Prize Collecting Steiner Tree 問題に対するヒューリスティクス

2015 年度修士論文

Prize Collecting Steiner Tree問題に対するヒューリスティクス

法政大学大学院 理工学研究科
システム工学 (経営系) 修士課程

14R6207　細川 祐樹

指導教員　五島 洋行

概 要

Prize Collecting Steiner Tree(PCST) 問題は，組合せ最適化の重要な問題である．PCST 問題は整数計画問題に定式化することで最適解を求めることができる．しかし整数計画問題の制約条件は入力の点数に比例して増加し，最適解を実時間内で求められないことがある．そのため，最適解を近似的に求める必要がある．

本研究は PCST 問題に対して，2 つのヒューリスティクス（H1, H2）を提案する．各ヒューリスティクスは，2 段階からなる．第 1 段階では，全域木を求める．第 2 段階では，全域木から目的関数を最小にする部分木を求める．部分木を求めるために新たに最適部分木問題を定義し，それを最適に解くアルゴリズムを用いる．

入力をランダムグラフと 5 つのグループ K, P, C, D, Cologne のベンチマークとし，計算実験を行った．計算実験の結果，2 つのヒューリスティクスは現実問題に基づいた K と Cologne の入力に対して，平均 1.3 未満の近似比を得た．これは P, C, D の入力と比べて，良好な結果である．サイズの大きな入力に対して，H1 は H2 と比べて高速に近似解を出力できるため，現実的な問題に適用できると考える．

目 次

1 はじめに **1**

2 Prize Collecting Steiner Tree 問題 **3**

3 PCST 問題に対する最適解の求め方 **4**

3.1 カットの概念 4

3.2 整数計画問題への定式化における制約条件の総数 5

4 比較対象となる近似アルゴリズム **7**

4.1 線形計画問題とプライマルデュアル法 7

4.2 PCST 問題に用いる主問題と双対問題 9

4.3 3-近似アルゴリズム 11

4.4 ダイクストラ法とクラスカル法 11

4.5 α 値の導出と 3-近似アルゴリズムの証明 13

4.6 主流とされる近似アルゴリズムの設計手法 15

4.7 2 近似アルゴリズム 15

4.8 2-近似アルゴリズムの証明 17

5 ヒューリスティクス (H1) **20**

5.1 ヒューリスティクス (H1) 20

5.2 グリーディな辺の選択 21

5.3 グリーディーアルゴリズムの解析 21

6 ヒューリスティクス (H2) **23**

6.1 有向グラフへの変換 23

6.2 最小コスト有向木の選択 24

6.3 多フェーズグリーディアルゴリズム 26

6.4 多フェーズグリーディーアルゴリズムの解析 27

6.5 ヒューリスティクス (H2) 28

7 最適部分木問題 **29**

7.1 最適な部分木を探索するアルゴリズム 29

7.2 アルゴリズムの正当性 29

7.3 計算時間の解析 31

8 計算実験 **32**
8.1 点数を固定したランダムグラフに対する結果 32
8.2 入力サイズの大きいランダムグラフに対する結果 34
8.3 K, P グループに対する結果 35
8.4 C, D グループに対する結果 37
8.5 光ファイバーの設置計画に基づいた入力に対する結果 40

9 考察 **42**

10 おわりに **43**

参考文献 **44**

謝辞 **45**

1　はじめに

電気供給が必要な家庭へ電線ケーブルを新設したり，いくつかの地点を結ぶようにパイプラインを設置する場合，できるだけ設置のコストを抑え，各地点を連結したいという要求が生じる．こういった要求はヨーロッパで良くみられ，通信回線，内陸水路のネットワークを強化し，ヨーロッパ内での貨物や旅客輸送の円滑化，効率化が進められている．その強化策の 1 つに TNT-T[1](Trand-European Transport Network) がある．この政策は欧州連合 (EU) 間のインフラ整備を整えることを目的としている．現在 EU の領域内には，交通道路がおよそ 500 万 km 以上，鉄道路線が 21 万 km 以上，水路が 4 万 km 以上あり，それらを効果的に活用するため，30 ほどの計画が進められている．その予算は 2007 年から 2013 年にかけておよそ 800 億円になる．したがって，それらの要求を解決することは重要であるといえる．

これを平面上に点が配置し，それらを結ぶ辺にコストを付随させ，全体を連結することを考える．これは最小全域木問題となり，標準的な解法で効率良く解ける．全体の点ではなく要求点のみを連結することを考えると，それは Steiner Tree 問題となる．また運輸や鉄道では，ある地点から出発し，同じ地点へ戻ってくることも考えられる．すべての点を一度のみ到達すると考えると，それは巡回セールスマン問題（TSP）となる．そして本研究で取り上げるのは Prize Collecting Steiner Tree(PCST) 問題である．PCST 問題は特別な点が 1 つ与えられ，点にペナルティが付随することが特徴の 1 つで，電線ケーブルや水道管の設置などのインフラ整備の効率化に対して応用が可能である．PCST 問題は Steiner Tree 問題を一般化したものであり，Steiner Tree 問題は NP 困難であることが証明されているため，PCST 問題も NP 困難である．

PCST 問題は NP 困難であるが，整数計画問題に定式化し最適解を求められる [1]．しかし整数計画問題の制約条件は入力の点数に比例して増加し，最適解を実時間内で求められないことがある．そのため，最適解を近似的に求める必要がある．

最適解を近似する代表的な手法にラウンディング法とプライマルディアル法がある．それらは PCST 問題のような組合せ最適化問題の近似アルゴリズム設計で良く用いられている．PCST 問題にもラウンディング法を用いて近似解を求める手法が存在し，近似比 3 を保証している [1]．プライマルデュアル法に基づいた近似アルゴリズムもある [2]．この近似アルゴリズムは近似比 2 を保証している．さらに近似比 1.9672 を保証する近似アルゴリズムが提案されている [3]．この近似アルゴリズムは，近似比が最も小さいことが知られている．最近では，メタヒューリスティクスのアプローチを取り入れた近似アルゴリズムが提案されている [4], [5].

本研究は PCST 問題に対して，2 つのヒューリスティクス（H1, H2）を提案し，計算実

[1]TNT-T: http://ec.europa.eu/transport/infrastructure/tentec/tentec-portal/site/en/abouttent.htm

験の結果から，PCST 問題を短時間で近似的に解く手法を検討する．

各ヒューリスティクスは，2 段階からなる．第 1 段階では，全域木を求める．H1 はグリーディアルゴリズムに従い，局所的に最適な選択を繰り返す．グリーディアルゴリズムとは，定めた基準に基づいて各ステップごとで局所的に最適化するようなアルゴリズムである [2]．H2 は，まず本研究で提案するアイデアを用いて，入力の無向グラフを有向グラフに変換する．次に変換された有向グラフから多フェーズグリーディアルゴリズム [6] に従って，全域木を求める．第 2 段階では，H1, H2 ともに第 1 段階で構築した全域木から目的関数を最小にする部分木を求める．そのため，新たに最適部分木問題を定義し，それを最適に解くアルゴリズムを用いる．

入力はランダムグラフと 5 つのグループ K, P, C, D, Cologne[2] のベンチマークとする．K は地図の構造に良く似ているが，P の構造はランダムである．C, D は OR-Library[3] に存在する入力に基づいて，新たに生成されている．Cologne は現実問題である光ファイバーの設置計画に基づいている．入力がランダムグラフのとき，ヒューリスティクスの性能を比較する対象に既存の 3-近似アルゴリズムを用いる [1]．計算実験から，H1, と H2 について，近似比の精度や計算時間について考察する．

[2]K, P, C, D, Cologne: http://dimacs11.cs.princeton.edu/downloads.html.
[3]OR-library: http://people.brunel.ac.uk/ mastjjb/jeb/info.html.

2 Prize Collecting Steiner Tree 問題

Prize Collecting Steiner Tree (PCST) 問題を以下に示す．無向グラフ $G = (V(G), E(G))$，各辺 e にコスト $c_e \geq 0$，各点 i にペナルティ $\pi_i \geq 0$，特別な点 $r \in V(G)$ を与える．点 r を含む木を T とするとき，以下の目的関数

$$\sum_{e \in E(T)} c_e \quad + \quad \sum_{i \in V(G) \setminus V(T)} \pi_i \,, \tag{2.1}$$

を最小にする木 T を求めたい [1].

また点集合 T に対する変数 Z_T とする．各 Z_T は最適な木に含まれないような点集合が選ばれ，1 に設定される．そのとき以下の目的関数

$$\sum_{e \in E(G)} c_e x_e + \sum_{T \subseteq V(G); r \notin T} \left(Z_T \sum_{v \in T} \pi_v \right) , \tag{2.2}$$

を最小にする木 T を求めたい

上記の目的関数を表す式 (2.1) と式 (2.2) は同意義なことが知られ [2]，式 (2.2) は近似アルゴリズムの基本設計の 1 つであるプライマルディアル法に適用される．

PCST 問題の入力例を図 1(a) に示す．二重丸の点は根 r，各点にペナルティ，各辺にコストが付随している．黒点のペナルティは 0 である．その入力に対して，実行可能な木を図 1(b) に示す．目的関数の値は 10 で，入力に対して最適な木 T である．

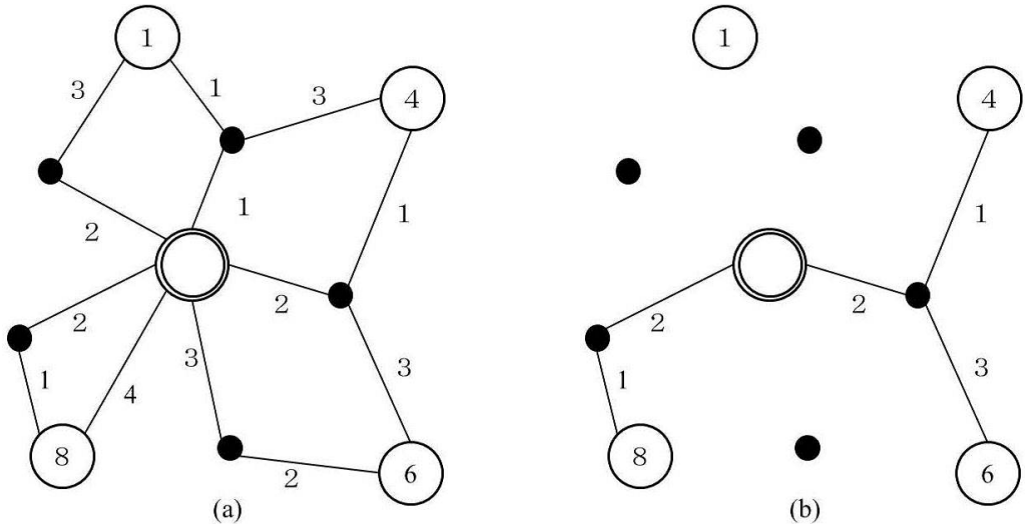

図 1　PCST 問題の入力例．入力 (a)，入力に対する最適な木 T (b)．二重丸の点は根 r，各点にペナルティ，各辺にコストが付随している．黒点のペナルティは 0.

3　PCST 問題に対する最適解の求め方

PCST 問題の入力を整数計画問題として定式化する詳細を示す．入力の辺と点に対応するようなバイナリ変数 $x_e\,(e \in E(G))$，$y_i\,(i \in V(G))$ を導入する．

$$x_e = \begin{cases} 1 & e \in E(T) \\ 0 & \text{otherwise} \end{cases}$$

$$y_i = \begin{cases} 1 & i \in V(T) \\ 0 & \text{otherwise} \end{cases}$$

変数 x_e，y_i の導入によって，以下のように整数計画問題に定式化することができる．

$$\text{目的関数} \quad \sum_{e \in E(G)} c_e x_e + \sum_{i \in V(G)} \pi_i (1 - y_i) \longrightarrow \min \tag{3.1}$$

$$\text{制約条件} \quad \sum_{e \in \delta(S)} x_e \geq y_i \qquad (S \subset V(G) \setminus \{r\},\ S \neq \emptyset,\ i \in S) \tag{3.2}$$

$$y_r = 1, \tag{3.3}$$

$$y_i \in \{0, 1\} \quad (i \in V(G) \setminus \{r\}) \tag{3.4}$$

$$x_e \in \{0, 1\} \quad (e \in E(G)) \tag{3.5}$$

$\delta(S)$ は，$V(G) \setminus \{r\}$ を S でカットした辺の集合である．PCST 問題の入力について，上記の定式化を用いて最適解を求める．

3.1　カットの概念

入力の点から任意にいくつかの点を選び，選んだ点の集合を K とする．入力の点は K とそれ以外の点で分けられ，このことを K によってカットするという．さらに K とそれ以外の点の要素で結ばれる辺を K によってカットされた辺といい，$\delta(K)$ を K でカットされた辺の集合とする．入力を図 2(a) とし，K に含まれる点を 3, 6 とする．このとき $\delta(K)$ は図 2(b) に示した i, ii, iii に相当する．

カットの概念は，グラフアルゴリズム理論において重要な考え方の 1 つである．カットの定義を用いることで，サイクルがなく連結な木を探索することができる．

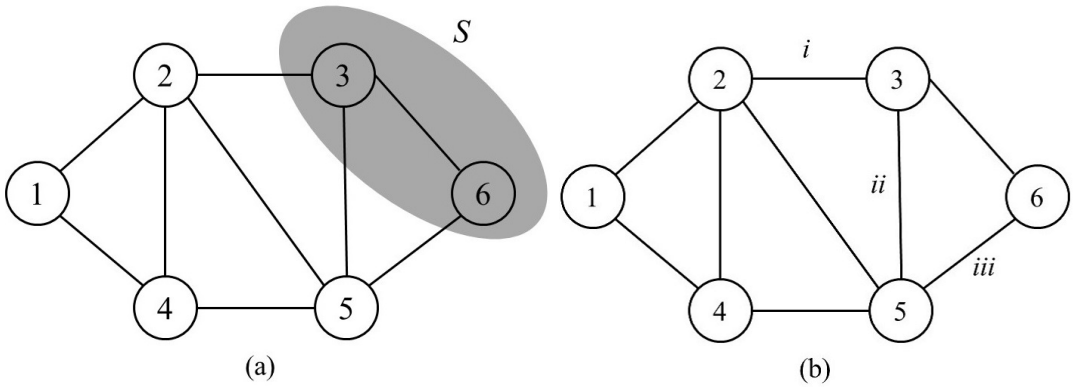

図 2　カットの概念図．入力と点の集合 S (a)，$\delta(S)$ に含まれる i, ii, iii (b)，各点に番号が与えられている．

3.2　整数計画問題への定式化における制約条件の総数

前述したように PCST 問題は，入力を整数計画問題に定式化し，最適解を求められる．定式化した際，最も重要な制約条件は式 (3.2) である．PCST 問題の出力は木であり，サイクルがなく，連結であるという条件が必要である．その条件の役割を果たすのが式 (3.2) である．これはグラフアルゴリズム理論に基づいたカットの概念用いている．カットを用いた制約は複雑であるため，その制約条件の総数に着目する．

仮定として入力の点数を n とする．S は $V(G) \setminus \{r\}$ の部分集合であり，空集合でない．したがって S の考えられる組合せは，

$$\sum_{i=1}^{n-1} \binom{n-1}{i}, \tag{3.6}$$

通りである．さらに S に含まれる点それぞれに制約条件が必要である．そのため，式 (3.2) の総和は，

$$\sum_{i=1}^{n-1} \binom{n-1}{i} \cdot i, \tag{3.7}$$

である．

点数と式 (3.2) の総数の対応関係を表 1 に示す．横軸は左から点数，式 (3.2) の総数である．点数 10 のとき，整数計画問題の制約条件の総数は 2.3×10^3 を超える．式 (3.2) のほかに，式 (3.3)，式 (3.4)，式 (3.5) の制約条件が存在するが，制約条件の総数はほとんどを式 (3.2) で占める．したがって整数計画問題に定式化した際，点数に比例して制約条件の総数は指数関数的に増える．そのため最適解を求めるために，非商用ソルバーの中で

表 1　入力の点数と制約条件の総数

点数	制約条件の総数	点数	制約条件の総数
10	2.3×10^3	60	1.7×10^{19}
20	5.0×10^7	70	2.0×10^{22}
30	7.8×10^9	80	2.4×10^{25}
40	1.1×10^{13}	90	2.8×10^{28}
50	1.4×10^{16}	100	3.1×10^{31}

も計算速度が高速といわれる SCIP ソルバーを用いる．しかし本研究の実験環境では，点数 15 以上になると最適解を求めることが難しくなる．

4　比較対象となる近似アルゴリズム

PCST 問題のような組合せ最適化問題は, 整数計画問題で定式化できるものが多い．そのように定式化されると，整数計画の緩和問題から最適解のコストを下から抑える自然な方法が得られる．最適解の下界を与えることは，近似アルゴリズムを設計する上で重要なカギである．緩和問題の実行可能解は，元の整数計画問題の小数解であり，その小数解は元の整数計画問題の実行可能解ではない．したがって問題は緩和問題の最適解を求めるのではなく，最適解に近い整数解を求めることである．

緩和問題を用いて，近似アルゴリズムを得るための 2 つの基本技法がある．1 つ目は、LP ラウンディングあるいはラウンディングと呼ばれる方法である．この方法は、より直接的方法であり，線形計画緩和の問題を解いて導いた小数解を，コストがそれほど増加しないように整数解に替えるというものである．近似保証は，小数解と整数解との比をとって決定される．PCST 問題をラウンディング法を用いて近似解を求める手法に，近似比 3 を保証するアルゴリズムがある [1].

2 つ目は，プライマルデュアル法と呼ばれる方法であり，ラウンディング法と比べると間接的でより複雑である．その方法は主問題を緩和問題として，主問題と双対問題の解をある条件を満たすように更新していく．双対問題の実行可能解は主問題の最適値 (OPT) に対する下界を与える．プライマルデュアル法に基づいた PCST 問題を解く手法も存在する [2]．この近似アルゴリズムは近似比 2 を保証している．

4.1　線形計画問題とプライマルデュアル法

前述したように，プライマルデュアル法は近似アルゴリズム設計の代表的手法である．この理論は，線形計画問題 (LP) に関連して構築される [2]．ここで，線形計画問題の基礎概念とプライマルデュアル法について示す．LP は，線形関数を線形不等式の制約式の下で最適化する問題である．ここで以下のような標準形の最小化問題を主問題とする．

$$\text{目的関数} \quad \sum_{j=1}^{n} c_j x_j \longrightarrow \min \tag{4.1}$$

$$\text{制約条件} \quad \sum_{j=1}^{n} a_{ij} x_j \geq b_i \quad (i = 1, 2, \cdots, m) \tag{4.2}$$

$$x_j \geq 0 \quad (j = 1, 2, \cdots, n) \tag{4.3}$$

標準形の最小化問題に対して，双対問題と呼ばれる問題は以下の線形計画問題である．

$$\text{目的関数} \quad \sum_{i=1}^{m} b_i y_i, \longrightarrow \max \tag{4.4}$$

$$\text{制約条件} \quad \sum_{i=1}^{m} a_{ij} y_i \leq c_j \quad (j = 1, 2, \cdots, n) \tag{4.5}$$

$$y_i \geq 0 \quad (i = 1, 2, \cdots, m) \tag{4.6}$$

双対問題の構成法から主問題と双対問題の最適解は一致することが知られている．また双対問題の実行可能解は主問題の最適値の下界を与え，その逆もいえる．したがって，主問題と双対問題それぞれに対する実行可能解で目的関数の値が一致するとき，それがともに最適解であることがいえる．このことを LP-双対定理という．また LP-双対定理より，主問題と双対問題の解 $\boldsymbol{x}$ と $\boldsymbol{y}$ がともに最適解であるための必要十分条件は，

$$x_j = 0 \quad \text{または} \quad \sum_{i=1}^{m} a_{ij} y_i = c_j \quad (1 \leq j \leq n), \tag{4.7}$$

$$y_i = 0 \quad \text{または} \quad \sum_{j=1}^{n} a_{ij} x_{j\_} b_i \quad (1 \leq i \leq m), \tag{4.8}$$

を満たすことである．主相補条件は式 (4.7)，双対相補条件は式 (4.8) のことを指す．この相補条件は効率的アルゴリズム設計において重要な役割を担う．プライマルディアル法は式 (4.7), 式 (4.8) を緩和し，以下のように用いる．

$$x_j = 0 \quad \text{または} \quad \frac{c_j}{\alpha} \leq \sum_{i=1}^{m} a_{ij} y_i \leq c_j \quad (1 \leq j \leq n) \tag{4.9}$$

$$y_i = 0 \quad \text{または} \quad b_i \leq \sum_{j=1}^{n} a_{ij} x_j \leq \beta \cdot b_i \quad (1 \leq i \leq m) \tag{4.10}$$

ここで新たにパラメータ $\alpha \geq 1$，$\beta \geq 1$ を導入する．$\boldsymbol{x}$ と $\boldsymbol{y}$ がともに実行可能解と仮定すると式 (4.9) より

$$\sum_{j=1}^{n} c_j x_j \leq \quad \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha \, a_{ij} \, x_j \, y_i \,, \tag{4.11}$$

がいえる．さらに式 (4.10) より，

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\alpha\, a_{ij}\, x_j\, y_i \le \sum_{i=1}^{m}\alpha\,\beta\, b_i\, y_i\,, \tag{4.12}$$

$$\sum_{j=1}^{n} c_j\, x_j \quad \le \alpha\cdot\beta\sum_{i=1}^{m} b_i\, y_i\,, \tag{4.13}$$

がいえる．

したがって，主問題の実行可能解は双対問題の実行可能解で算出される評価値の $\alpha\cdot\beta$ 倍以下であることがいえる．プライマルデュアル法を適用したアルゴリズムは初期値に自明な解 $\boldsymbol{x}=0$ と $\boldsymbol{y}=0$ を用いる．それは $\boldsymbol{x}$ と $\boldsymbol{y}$ がともに実行可能解であることを仮定したためである．そして主問題と双対問題の解の改善を各反復で達成し，最終的に主問題の実行可能解を得ると同時に，適切に選んだ α と β について，両方の問題の解が式 (4.7)，式 (4.8) をすべて満たすようにする．主問題の解は常に整数の値をとりながら更新され，最終的に得られた解も整数解となる．

4.2 PCST 問題に用いる主問題と双対問題

PCST 問題における主問題と双対問題を示す [7]．各辺 e に対する変数 x_e と各点集合 T に対する変数 Z_T を導入する．各 Z_T は最適な木に含まれないような点集合が選ばれ，1 に設定する．

$$\text{目的関数}\quad \sum_{e\in E(G)} c_e x_e + \sum_{T\subseteq V(G);r\notin T}\left(Z_T\sum_{v\in T}\pi_v\right)\longrightarrow \min \tag{4.14}$$

$$\text{制約条件}\quad \sum_{e\in\delta(S)} x_e + \sum_{T\supseteq S} Z_T \ge 1 \quad (S\subseteq V(G)\setminus\{r\}; S\neq\emptyset) \tag{4.15}$$

$$x_e\in\{0,1\}\quad (e\in E(G)) \tag{4.16}$$

$$Z_T\in\{0,1\}\quad (T\subseteq V(G)\setminus\{r\}) \tag{4.17}$$

整数計画問題の解は，0 か 1 の値をとるため制約が厳しい．解が 0 以上をとれるよう緩和することを，これを整数計画問題を緩和するといい，制約を緩めることがある．変更点はバイナリ変数 x_e, Z_T を $x_e\ge 0, Z_T\ge 0$ にする点である．整数計画の緩和問題を以下のように定式化する．

$$\text{目的関数} \quad \sum_{e\in E(G)} c_e x_e + \sum_{T\subseteq V(G);r\notin T} \left(Z_T \sum_{v\in T} \pi_v \right) \longrightarrow \min \tag{4.18}$$

$$\text{制約条件} \quad \sum_{e\in\delta(S)} x_e + \sum_{T\supseteq S} Z_T \geq 1 \quad (S \subseteq V(G)\setminus\{r\}; S \neq \emptyset) \tag{4.19}$$

$$x_e \geq 0 \quad (e \in E(G)) \tag{4.20}$$

$$Z_T \geq 0 \quad (T \subseteq V(G)\setminus\{r\}) \tag{4.21}$$

この問題を PCST 問題の主問題とする．主問題があるとき，その双対問題が存在する．双対問題とは，実行可能解が主問題の下界を与え，双対問題の最適解と主問題の最適解は等しくなる．以下に主問題に対する双対問題を示す．双対変数 $y_S(S \subseteq V(G)\setminus\{r\}; S \neq \emptyset)$ を導入する．

$$\text{目的関数} \quad \sum_{S\subseteq V(G)\setminus\{r\};S\neq\emptyset} y_S \longrightarrow \max \tag{4.22}$$

$$\text{制約条件} \quad \sum_{S\subseteq V(G)\setminus\{r\};e\in\delta(S)} y_S \leq c_e \quad (e \in E(G)) \tag{4.23}$$

$$\sum_{S\subseteq T;S\neq\emptyset} y_S \leq \sum_{i\in T} \pi_i \quad (T \subseteq V(G)\setminus\{r\}) \tag{4.24}$$

$$y_S \geq 0 \quad (S \subseteq V(G)\setminus\{r\}; S \neq \emptyset) \tag{4.25}$$

$\delta(S)$ は，$V(G)\setminus\{r\}$ を S でカットした辺の集合である．

次に主問題と双対問題からある条件を示す．$\boldsymbol{x}$ と $\boldsymbol{y}$ を主問題と双対問題の実行可能解とする．このとき $\boldsymbol{x}$ と $\boldsymbol{y}$ が最適解であるための必要十分条件に，

$$x_e = 0 \quad \text{または} \quad \sum_{S\subseteq V(G)\setminus\{r\};e\in\delta(S)} y_S = c_e \quad (e \in E(G)), \tag{4.26}$$

$$Z_T = 0 \quad \text{または} \quad \sum_{S\subseteq T;S\neq\emptyset} y_S = \sum_{i\in T} \pi_i \quad (T \subseteq V(G)\setminus\{r\}), \tag{4.27}$$

$$y_S = 0 \quad \text{または} \quad \sum_{e\in\delta(S)} x_e + \sum_{T\supseteq S} Z_T = 1 \quad (S \subseteq V(G)\setminus\{r\}; S \neq \emptyset), \tag{4.28}$$

を満たすことがいえ，主相補条件は式 (4.26), 式 (4.27)，双対相補条件は式 (4.28) といえる．

したがって PCST 問題の主問題を式 (4.1)，式 (4.2)，式 (4.3) とすると，その双対問題から主相補条件と双対相補条件が導かれる．

4.3 3-近似アルゴリズム

3-近似アルゴリズムは単純なラウンディングアルゴリズムを適用している．はじめに前述に示した整数計画の緩和問題に定式化する．その緩和問題の解を $\boldsymbol{x}^*$ と $\boldsymbol{y}^*$ とする．次に U を条件 $y_i^* \geq \alpha$ を満たす点 i の集合とする．パラメータ α は適切な値に定められ，PCST 問題の α 値は $\alpha = \frac{2}{3}$ である．α 値の導出と 3-近似アルゴリズムの証明は後に示す．次に，U の要素をすべて含む木を構築する．その木は条件を満たす中で，最小コストの木とする．最小コストの木は，辺のコストの総和が最小となる木を指し，構築された最小コストの木を木 T として出力する．

3-近似アルゴリズム

1. 入力を整数計画の緩和問題に定式化し，その解を $\boldsymbol{x}^*$ と $\boldsymbol{y}^*$ とする．
2. $U \longleftarrow \{i \in V(G) : y_i^* \geq \alpha\}$ ただし $\alpha = \frac{2}{3}$.
3. すべての $i(\in U)$ を含む，できるだけ最小コストの木 T を求める．

4.4 ダイクストラ法とクラスカル法

3-近似アルゴリズムを実装する際に，本研究で扱う主なアルゴリズムを示す．これらのアルゴリズムは 3-近似アルゴリズムにおけるステップ 3 で用いる．ステップ 3 はある 2 つの問題を解く必要がある．1 つ目が最短経路問題である．ステップ 2 で選ばれた点について，各点の最短経路を求める．2 つ目が最小コスト全域木問題である．ステップ 2 で選ばれた点について，それぞれ最短経路を求めた後，それらの点をすべて含む最小コストの木を求める．最短経路問題はダイクストラ法，最小コスト全域木問題はクラスカル法を用いる．以下に問題の定義とアルゴリズムの詳細を示す．

- **最短経路問題**

有向グラフ $G = (V(G), E(G))$ と始点 s が与えられる．なお G は s からほかのすべての点への経路をもつものと仮定する．各辺 e には長さ $l_{e \in S} \geq 0$ が付随する．経路 $P \in E(G)$ に対して，P の長さは，P に含まれるすべての辺の長さの合計であり，$l(P)$ と表す．このとき，s からすべての点の最短パス $l(P)$ を求めることが問題である．

最短経路問題は，有向グラフを対象としている．無向グラフの場合，辺 (u, v) をそれぞれ枝 (u, v) と枝 (v, u) の有向枝に変換し，最短経路問題を解くアルゴリズムを用いる．つまり入力が無向グラフの PCST 問題にも適用できる．この問題は，始点 s からほかのすべての点に対して最短経路を導くダイクストラ法を用いる．

ダイクストラ法

ダイクストラ法は単一始点問題，つまり，始点 s からグラフのほかのすべての点に対して最短経路を導くアルゴリズムであり，単純なグリーディアルゴリズムである．ダイクストラ法を以下に示す．

1. $S = \{s\}, d(s) = 0$ とする．
2. $|S| = |V(G)|$ を満たすまで以下の作業を繰り返す．
   少なくとも S からの辺をもつ点 $v \in V(G)$ の中で，$d'(v) = \min(d(u) + l_{e \in S})$ が最小となる点 v を選ぶ．
3. 点 v を S に追加し，$d(v) = d'(v)$ とする．

S は確認済み点の集合であり，各 $u \in S$ に対して，s からの距離を $d(u)$ とする．点 $v \in V(G) \setminus S$ に対して，集合 S の確認済み点を経由し，点 v に最短で到達する経路を求める．点 v を S に加え，$d(v)$ の値を $d'(u)$ と確定する．

- 最小コスト全域木問題

全域木とはサイクルがなく連結であり，入力の点をすべて含む木のことである．つまり，どの 2 点間にも経路が存在する．与えられた入力において，コストが最小の全域木を求める問題を最小コスト全域木問題という．最小全域木を求めるアルゴリズムは多数存在するが，本研究はクラスカル法を用いる．

クラスカル法

1. リスト $F = \emptyset, F' = \emptyset$ とする．
2. 枝 e をコストが小さい順にリスト F へ加える．
3. 次の条件に従い，リスト F から枝 e を逐次的に取り出し，F' に加える．

   F' で構成されるグラフについて，枝 e を F' に加え，サイクルがないとき，枝 e を F' に加える．一方，枝 e を加えることで，サイクルが生じる場合，枝 e は選ばず，リスト F から次の枝を取り出す．F' がすべての 2 点間の経路が存在したとき終了し，そのとき F' は最小コスト全域木である．

クラスカル法は，グリーディアルゴリズムの一つであり，構築される全域木は最小全域木であることが保証されている．

4.5 α 値の導出と 3-近似アルゴリズムの証明

3-近似アルゴリズムの実装のために，α 値の導出と 3-近似アルゴリズムの証明をする必要がある．以下に 2 つの命題を示す．はじめに，最小コストを求めるアルゴリズムの性質から以下の式が成り立つ．

補題 4.1

$$\sum_{e \in T} c_e \leq \frac{2}{\alpha} \sum_{e \in E(G)} c_e x_e^* \tag{4.29}$$

補題 4.1 は，本章で扱う最小コストの木を求めるアルゴリズムとは異なるアルゴリズムを用いて，証明できることが知られている [2]．そのため本章では，式 (4.29) の証明は割愛する．ただし 2 つのアルゴリズムの出力はどちらも，条件を満たす最小コストの木であり，本章で扱うアルゴリズムも式 (4.29) を満たすことがいえる．

補題 4.2

$$\sum_{i \in V(G) \setminus V(T)} \pi_i \leq \frac{1}{1-\alpha} \sum_{i \in V(G)} \pi_i (1 - y_i^*) \ 2 \tag{4.30}$$

点 i が木 T に含まれないことは，点 i が点集合 U に含まれないことと同意義であり，そこから $y_i < \alpha$ が成り立つ．したがって $1 - y_i^* > 1 - \alpha$ より $\frac{1-y_i^*}{1-\alpha} > 1$ である．よって

$$\sum_{i \in V(G) \setminus V(T)} \pi_i \leq \sum_{i \in V(G)} \pi_i \,, \tag{4.31}$$

が $V(G) \setminus V(T) \subseteq V(G)$ と $\pi_i \geq 0$ が成り立つことにより，補題 4.2 は自明であるため，式 (4.30) が成り立つことがいえる．そして，2 つの命題から以下の式が成り立つ．

定理 4.3

$$\sum_{e \in T} c_e + \sum_{i \in V(G) \setminus V(T)} \pi_i \leq \frac{2}{\alpha} \sum_{e \in E(G)} c_e x_e^* + \frac{1}{1-\alpha} \sum_{i \in V(G)} \pi_i (1 - y_i^*) \tag{4.32}$$

ラウンディングアルゴリズムは緩和問題の解を用いて上界を求め，その値が小さいほど性能の良いアルゴリズムといえる．定理 4.3 の右辺はパラメータ α の値に依存する．そのため，$\max\{\frac{2}{\alpha}, \frac{1}{1-\alpha}\}$ を最小とする α 値を求める．$\frac{2}{\alpha} = \frac{1}{1-\alpha}$ を解くと，$\alpha = \frac{2}{3}$ のとき，条件 $\max\{\frac{2}{\alpha}, \frac{1}{1-\alpha}\}$ が最小値になり，以下の式が成り立つ．

性質 4.4

$$\sum_{e\in T} c_e + \sum_{i\in V(G)\setminus V(T)} \pi_i \leq 3\left(\sum_{e\in E(G)} c_e x_e^* + \sum_{i\in V(G)} \pi_i(1-y_i^*)\right) \leq 3\ \mathrm{OPT} \tag{4.33}$$

性質 4.4 の左辺は出力する木 T の評価値であり，その値が入力の最適値 OPT の 3 倍以下ということを示している．このことを近似保証 3 といい，そのアルゴリズムを 3-近似アルゴリズムという．以下の

$$\sum_{e\in E(G)} c_e x_e^* + \sum_{i\in V(G)} \pi_i(1-y_i^*)\,, \tag{4.34}$$

は緩和問題の最適解である．緩和問題とその双対問題の関係から

$$\sum_{e\in E(G)} c_e x_e^* + \sum_{i\in V(G)} \pi_i(1-y_i^*) \leq \sum_{e\in E(G)} c_e x_e + \sum_{i\in V(G)} \pi_i(1-y_i) = \mathrm{OPT}\,, \tag{4.35}$$

が成り立つため，本章で示した 3-近似アルゴリズムの近似保証 3 が証明される．

4.6　主流とされる近似アルゴリズムの設計手法

現在，近似アルゴリズム設計で主流とされるのはプライマルデュアル法であり，その手法を取り入れているのが 2-近似アルゴリズムである．前述した 3-近似アルゴリズムとの大きな違いは，主問題と双対問題の解を繰り返し更新する点である．そのため整数計画の緩和問題を解く必要がない．前述した主相補条件と双対相補条件を満たしさえすれば，ある一定のルールに基づいて多項式時間に近似解を出力することが可能である．

4.7　2 近似アルゴリズム

2-近似アルゴリズムの詳細を以下の用語を用いて説明する．y_S は双対問題の変数とし，S は点の集合に相当する．

- 同期する (synchronized manner)：y_S の値を同時に増加させること．
- 持ち上げる (raised)：$y_S \neq 0$ のとき，集合 S は y_S の値で**持ち上げられている**という．
- 触れている (feel)：$y_S \neq 0$ かつ $e \in \delta(S)$ であるとき e は y_S で **触れている**という．
- タイト (tight), オーバータイト (overtight)：e が**触れている** y_S の総和がコスト c_e に等しいとき，**タイト**であるという．すなわち，y_S の総和がコスト c_e より大きいとき，**オーバータイト**であるという．
- 活性集合 (active set)：活性集合 S と対応する y_S は**同期**をもって増加される．
- チャージ (charge)：y_S を増加させることができる最大量．
- 死亡 (dead)：y_S の**チャージ**がなくなること．
- 実行可能 (feasible)：r のマークが付いた各点から，r への辺の集合 F からなるパスが存在するとき，**実行可能**であるという．
- 極大死亡集合 (maximal dead set)：死亡と宣言された集合で，要素数が最大の集合．
- e に関する v を含む極大死亡集合 (maximal dead set w.r.t. e, containing v)：
  $e = (u, v)$ に着目し，死亡と宣言された S が $v \in S, u \notin S$ または $u \in S, v \notin S$ の条件を満たした極大の集合 S であるとする．このとき，S を $e = (u, v)$ に関する v もしくは u を含む**極大死亡集合**という．

これらの用語はグラフアルゴリズム理論で良く扱われている．特に 2-近似アルゴリズムはチャージの概念が重要であり，近似解は各点にチャージした総和をとる．そのため主相補条件と双対問題を満たし，どれだけチャージできるかを追求する．そのような動作を繰り返すため，2-近似アルゴリズムは多項式時間で実装ができると知られている．

2-近似アルゴリズム

- 初期化

1. 各点 $v \neq r$ が単一点からなる活性集合でチャージ π_v を持つ．
2. 順序付きリスト F を $\emptyset$ とする．
3. すべての活性集合は同期をとって増加する．
4. すべての点をマークなしとする．

- 活性集合がなくなるまで 1，2 を繰り返す

1. 各活性集合 S に対応する y_S を，いずれかの e がタイトになるまで持ち上げる．ただし集合 S のチャージがなくなるとき，S を死亡とし，S に含まれるマークのついていない点を “S” とする．
2. e がタイトになるとき，リスト F に加える．
   - e が活性集合 S と直接 r を結ぶとき，S を不活性とし，r に連結してるとする．S に含まれるマークのついていない点を “r” とする．
   - e が活性集合 S と r に連結された集合を結ぶとき，上記と同様の動作を行う．
   - e がともに活性集合であるか，一方が活性集合で他方が死亡の集合となる S と S' を結ぶとき，どちらも不活性集合とし，さらに $S \bigcup S'$ を新たな活性集合とする．その活性集合のチャージは S と S' に残っているチャージの和とする．

- 動的逆順削除

1. “r” のマークが付く点を要求点とする．
2. e（$\in F$）について，リスト F に加えた順番と逆の順番の辺に着目する．
   - $F \setminus \{e\}$ が実行可能ならば，e をリスト F から除く．
   - そうでないとき $e = \{u, v\}$ とし，S を e に関する v もしくは u を含む極大死亡集合とする．$S \neq \emptyset$ ならば，S に属する点すべてを要求点とする．
   - e に関して，v もしくは u を含む極大死亡集合について，同様のことを繰り返す．
3. リスト F に加えた辺すべてに，同様のことを繰り返す．

4.8 2-近似アルゴリズムの証明

近似アルゴリズムによる近似解は最適値の 2 倍以下，つまり近似保証が 2 である．そのため，主問題と双対問題の解が

$$\sum_{e\in E(G)} c_e x_e + \sum_{T\subseteq V(G);r\notin T} \left(Z_T \sum_{i\in T} \pi_i \right) \leq 2 \sum_{S\subseteq V(G);r\notin S} y_S, \tag{4.36}$$

を満たすことを証明する．入力の Z_T がバイナリ変数，$\pi_i (i \in V(G)) \geq 0$ から，

$$\sum_{T\subseteq V(G);r\notin T} \left(Z_T \sum_{i\in T} \pi_i \right) \geq 0, \tag{4.37}$$

である．そのため，

$$\sum_{e\in E(G)} c_e x_e + 2 \sum_{T\subseteq V(G);r\notin T} \left(Z_T \sum_{i\in T} \pi_i \right) \leq 2 \sum_{S\subseteq V(G);r\notin S} y_S, \tag{4.38}$$

を示せば，式 (4.36) も成立する．したがって，アルゴリズムの近似保証 2 を式 (4.38) が成り立つことで示す．

補題 4.5

$$Z_T = 1 \ (r \notin T) \quad \text{ならば} \quad \sum_{i\in T} \pi_i = \sum_{S\supseteq T} y_S \tag{4.39}$$

補題 4.5 は $Z_T = 1$ ならば T に含まれる点のペナルティの和は，T に含まれる集合の双対変数の値の和に等しいことを指す．このことは帰納法によって証明できる．はじめ，入力は各単一点 $i(\in V(G) \setminus \{r\})$ からなる活性集合である．$Z_T = 1$ ならば，T に含まれる点はすべて r と連結でない．r と連結でないとき，各単一点集合 S は不活性集合でなく，死亡であると宣言される．死亡の定義は S のチャージがなくなることである．各単一点集合のチャージは各点のペナルティと等しく，双対変数の値も各点のペナルティと等しい．単一点集合 S が不活性集合と宣言されることもある．単一点集合 S と S' を結ぶ辺がタイトであるとき，どちらの単一点集合も不活性とし，さらに $S \bigcup S'$ を活性集合 S'' とする．その活性集合のチャージは，S と S' の残っているチャージの和である．S'' が最終的に死亡であると仮定したとき，それぞれの双対変数の和は

$$y_S \quad + \quad y_{S'} \quad + \quad y_{S''} \quad = \sum_{i \in S''} \pi_i, \tag{4.40}$$

となる．これは S と S' が単一集合でないときも成り立つ．ここで T に含まれる点について議論する．最終的に死亡と宣言された集合に含まれる点は T に属する．つまり，順次に点の重複がない極大死亡集合を選択し，それに含まれる点を T へ加えると同義である．式 (4.39) より，それぞれの極大死亡集合に含まれる双対変数の和は各点 i における π_i の和である．T に含まれる点が単一点でない集合 S に属するとき，双対変数 y_S は式 (4.40) を満たす，もしくは 0 であるため，式 (4.38) が成り立つことが証明される．よって式 (4.39) を用いると，

$$\sum_{e \in E(G)} c_e x_e \ + 2 \sum_{T \subseteq V(G); r \notin T} \left(Z_T \sum_{i \in T} \pi_i \right) \leq 2 \sum_{S \subseteq V(G); r \notin S} y_S, \tag{4.41}$$

$$\sum_{e \in E(G)} c_e x_e \ + 2 \sum_{i \in T; Z_T = 1} \pi_i \leq 2 \sum_{S \subseteq V(G); r \notin S} y_S, \tag{4.42}$$

$$\sum_{e \in E(G)} c_e x_e \ + 2 \sum_{S \supseteq T; Z_T = 1} y_S \leq 2 \sum_{S \subseteq V(G); r \notin S} y_S, \tag{4.43}$$

がいえる．さらに

$$\sum_{e \in E(G)} c_e x_e \leq \ 2 \sum_{S \subseteq V(G); r \notin S} y_S - 2 \sum_{S \supseteq T; Z_T = 1} y_S, \tag{4.44}$$

$$\sum_{e \in E(G)} c_e x_e \leq \ 2 \sum_{S \subseteq V(G) \setminus T; Z_T = 1} y_S, \tag{4.45}$$

であるため，式 (4.38) は式 (4.45) に展開できる．ここで以下に，$\deg_F(S)$ を定義する．

- $\deg_F(S)$：カット (S) に含まれる F の辺の数

この定義は，満たされる集合に次数を与えることを意味する．これは. 近似アルゴリズムだけでなく，様々なアルゴリズムの近似保証を示す際に決定的に重要な役割を果たす．F' を 2-近似アルゴリズムによる解の辺の集合と仮定すると

$$\sum_{e \in E(G)} c_e x_e = \sum_{e \in F'} c_e, \tag{4.46}$$

がいえる．x_e は $e \in F$ を満たすとき 1，それ以外のとき 0 である．よって式 (4.46) が成り立つことは自明である．

補題 4.6

$$\sum_{e \in F'} c_e = \sum_{e \in F'} \left(\sum_{S : e \in \delta(S)} y_S \right) \tag{4.47}$$

補題 4.6 はリスト F' に加えられた辺はすべてタイトなことを指す．ここで和の順番を交換すると

$$\sum_{e \in F'} c_e = \sum_{S \subseteq V(G)} \left(\sum_{F' \bigcap e \in \delta(S)} y_S \right) = \sum_{S \subseteq V(G)} \deg'_F(S) \cdot y_S, \tag{4.48}$$

が得られる．したがって以下の不等式を証明すれば良い．

$$\sum_{S \subseteq V(G)} \deg'_F(S) \cdot y_S \leq 2 \sum_{S \subseteq V(G)} y_S \tag{4.49}$$

各反復で F' に関する活性集合全体で点の平均次数が高々2 であることを式 (4.49) は指す．木あるいは森では一般的に点の平均次数が 2 であることが知られているため [2]，式 (4.48) が成り立つといえる．したがって，2-近似アルゴリズムの近似保証は 2 である．

5　ヒューリスティクス (H1)

提案するヒューリスティクス (H1) はグリーディアルゴリズムにしたがって目的関数の最小化を目指す．一般にグリーディアルゴリズムは何らかの基準に基づいて，各ステップごとで局所的に最適化し，作り上げるアルゴリズムである [2]．したがって同じ問題に対して，多くの異なるグリーディアルゴリズムが設計されることもある．

一方，自明でない問題をグリーディアルゴリズムで最適に解くことができるときは，一般にその問題の構造そのものに興味深く有益なものが存在しているといえる [2]．すなわち最適な解を構成するために利用できる局所的な決定規則が存在する．本研究で新しく定義する最適部分木問題に対して，最適な木を導くアルゴリズムもその 1 つである．最適な木を導くために，局所的な決定規則に従うアルゴリズムを用いる．最適部分木問題の定義と用いたアルゴリズムによって最適な部分木が出力される証明は後に示す．

グリーディアルゴリズムを作成することは簡単であるが，それが良好に機能し，かつ良好に機能することを証明することは難しい．

5.1　ヒューリスティクス (H1)

提案するヒューリスティクス (H1) を示す．ヒューリスティクスは大きく分けて 2 段階からなる．第 1 段階はグリーディアルゴリズムにしたがい全域木を求める．本研究で用いたグリーディアルゴリズムは，局所的かつ楽観的に判断を行うシンプルなものである．第 2 段階では，得られた全域木から最適な部分木を求める．以下に提案ヒューリスティクス (H1) を示す．

ヒューリスティクス (H1)

全域木の構築

1. 根を選択．
2. グリーディアルゴリズムに従い，辺を選択．
3. すべての点を結ぶまで，繰り返す．

点の削除

• H1 の辺の削除は最適な部分木を求めることと同意義であり，最適な部分木を求めるアルゴリズムを後に示す．そのアルゴリズムは入力が無向木のときも適用できる．

5.2 グリーディな辺の選択

グリーディアルゴリズムは組合せ最適化問題で良く用いるアルゴリズムであり，PCST問題にも適用する．以下にそのアルゴリズムを示す．

グリーディアルゴリズム

• 無向グラフ $G = (V(G), E(G))$ の全域木 $T = (V(T), F(T))$ の構築．各辺 e にコスト $c_e \geq 0$，各点 i にペナルティ $\pi_i \geq 0$，特別な点 $r \in V(G)$ が存在する．

1. リスト $F = \emptyset$, $F \leftarrow r$ とする．
2. $V(G)$ を F でカットしたときの辺集合 $\delta(F)$ から，条件 $\pi_{j \notin F} - c_{e \in \delta(F)}$ が最大の辺 e を 1 本選ぶ．
3. 2 で選んだ辺 e によって結ばれる点 j を F に加える．
4. $|F| = |V(G)|$ を満たすまで 2, 3 を繰り返す．

はじめに r をリスト F に加える．PCST 問題で求めたい木 T は r と連結という条件があるため，はじめに r を選ぶことは自然なことである．次に F で G をカットしたときの辺集合 $\delta(F)$ から，目的関数を最小とする辺を 1 本選ぶ．それは条件 $\pi_{j \in F} - c_{e \in \delta(F)}$ が最大の e を選ぶことと同意義である．そのとき選んだ e によって結ばれる点 j を F に加える．その概念図を図 3(a) に示す．F の点が黒い部分にすべて含まれるとき，$\delta(F)$ に相当するのは，s, t, u であり，条件 $\pi_{j \in F} - c_{e \in \delta(F)}$ を最大にする e が選ばれる．選ばれた辺を s と仮定すると，結ばれる点は図 3(b) の v_s である．したがって v_s を F に加える．この動作を F がすべての点を含むまで繰り返したとき，選んだ辺で構成されるのは G の全域木である．

5.3 グリーディーアルゴリズムの解析

入力の点数を n，枝数を m と仮定する．はじめの辺の選択肢は高々 m 通りで，辺が 1 つ選択されるごとに選択肢は 1 つ減る．選択する回数は木の性質から $n-1$ であるため，その計算時間は

$$m + (m-1) + \cdots + (m-n-1) = \frac{n(m+1)}{2}, \tag{5.1}$$

$$= \mathrm{O}(nm + n), \tag{5.2}$$

である．

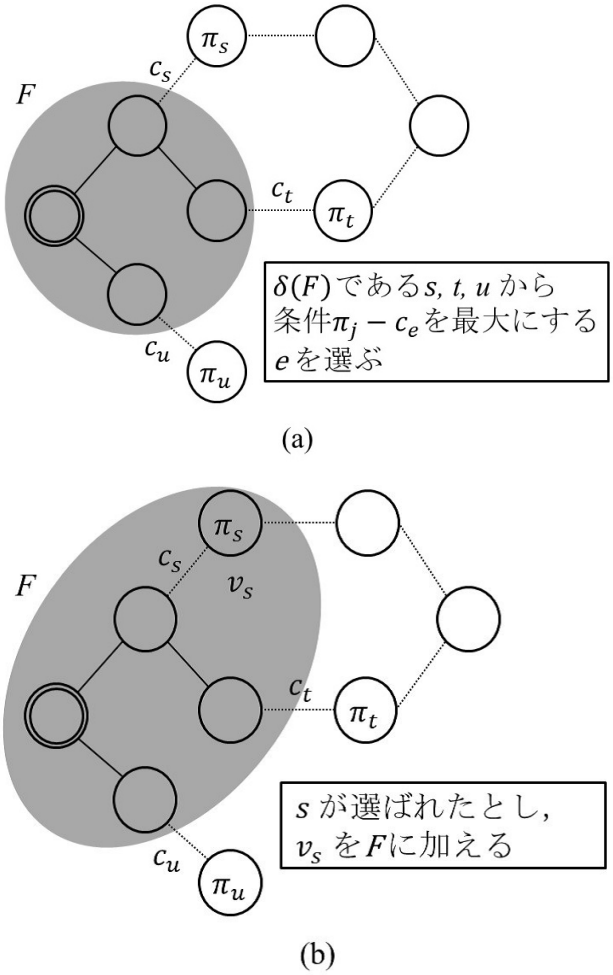

図 3　グリーディアルゴリズムの動作例．F と $\delta(F)$ に相当する辺 (a)，選ばれた辺を s としたとき F に加える点 v (b)．二重丸の点は根 r，点にペナルティ，辺にコストが付随している．

したがってグリーディアルゴリズムの時間計算量は $\mathrm{O}(nm + n)$ であり，多項式時間で実装が可能である．

6　ヒューリスティクス (H2)

前述したヒューリスティクス (H1) はグリーディアルゴリズムに従って全域木を求める．しかし，グリーディアルゴリズムは局所的かつ楽観的に判断するシンプルなアルゴリズムと述べた．それは，はじめに順序を決定して逐次的に決定しているためである．よって全体的な計画は事前にすべて明らかになっていない．そのアルゴリズムを用いた理由として，入力グラフの点に付随するペナルティの存在が挙げられる．ペナルティの存在を無視する，つまり辺に付随するコストのみで最小コストの全域木を考えるならば，それは最小コスト全域木問題である．これは前述したが，この問題も自然なグリーディアルゴリズムを用いて最小コストの全域木を求められ，本研究で用いたクラスカル法がその 1 つである．この方法はコストの昇順に辺を適切に選び，それを逐次的に行う．さらにプリム法や逆順削除アルゴリズムなど最適なアルゴリズムが多数存在する [2]．それらのアルゴリズムは多項式時間で解くことができる．よって，そこから入力で与えられるペナルティをコストに反映させるという自然なアイデアが生まれてくる．その際，入力の無向グラフを有向グラフに変換させ，その有向グラフに対して最小コスト有向木を求める．最終的に求めた最小コスト有向木を全域木とみなす．

6.1　有向グラフへの変換

無向グラフを有向グラフに変換するアイデアを提案する．このことは入力で与えられるペナルティをコストに反映させるという自然なアイデアから生まれている．無向辺を有向辺に変換するアイデアを説明するため，以下のような定義をする．

- i, j を結ぶ辺を x_{ij} とし，コストを c_{ij} とする．また x_{ij} と x_{ji} は等しい．
- i から j に向かう枝を x^*_{ij} とし，コストを c^*_{ij} と仮定する．また枝の性質より x^*_{ij} と x^*_{ji} は異なる．

一般的な辺 x_{ij} を図 4(a) に示す．i, j はそれぞれペナルティ π_i, π_j が付属し，それらを結ぶ辺のコストを c_{ij} とする．その x_{ij} を 2 本の有向枝に変換したのが図 4(b) である．有向枝 x^*_{ij} のコスト c^*_{ij} は $c^*_{ij} := c_{ij} - \pi_j$ と定義する．この動作を各辺にすることで，入力の無向グラフはペナルティを取り除いた有向グラフになる．この有向グラフは辺に付随するコストに対して，両端点に付随するペナルティが大きいほど小さい値をとる有向枝が存在する．

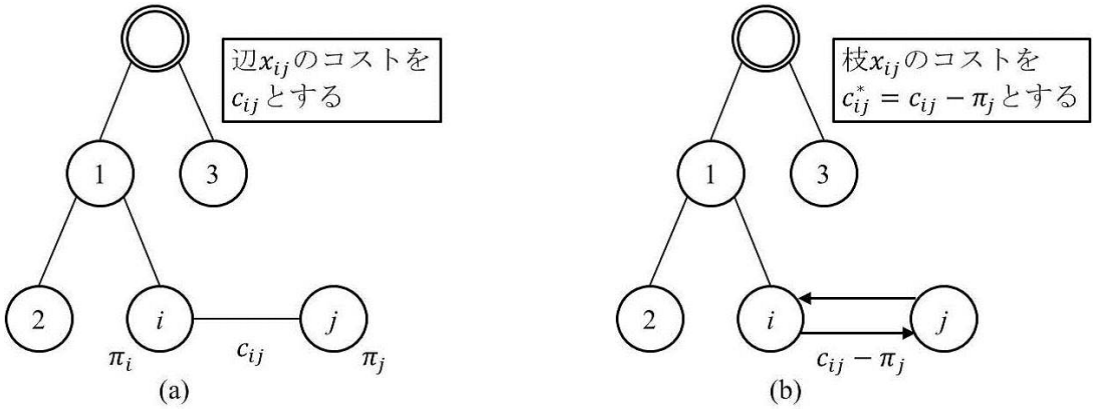

図 4　有向枝に変換するアイデア．一般的な無向辺の例 (a)，変換される 2 本の有向枝 (b)．二重丸の点は根 r，点にペナルティ，辺にコストが付随している．

6.2　最小コスト有向木の選択

有向グラフから最小コストの有向木を求める問題を最小コスト有向木問題といい，最適な木を出力するアルゴリズムが存在する [6]．最小コスト有向木問題の定義を以下に示す．有向グラフ $G = (V(G), E(G))$，各辺 e にコスト c_e，特別な点 $r \in V(G)$ を与える．点 r を根とする有向木を T とするとき，以下の目的関数

$$\sum_{e \in E(T)} c_e \,, \tag{6.1}$$

を最小にする全点有向木 T を求めたい [6]．

全点有向木 T とは，辺の向きを無視したとき G の全域木である．さらに辺の向きを考慮すると，ほかのすべての $v \in V(G)$ へのパスが存在するとき，T を G における全点有向木であるという．

最小コスト有向木問題の入力例を図 5(a) とする．二重丸の点は根である．その入力における最適な有向木 T は図 5(b) である．最小コスト有向木を求めるアルゴリズムは無向グラフにおける最小全域木問題の有向グラフ版と見なせる．しかし有向グラフにすることにより，新たな困難が生じる．最小全域木問題の解決法であるクラスカル法やプリム法と異なり，再帰的，逐次的に有向木を求める必要がある．最小コスト有向木問題の解決法は多フェーズグリーディアルゴリズムと呼ばれる．多フェーズグリーディアルゴリズムはグリーディな形式をとっているが，枝を選ぶ規則が少し洗練されたものである．コストを昇順に加えていくクラスカル法では図 5(a) において，最適な有向木を求められない．多フェーズグリーディアルゴリズムは図 5(a) のような入力においても，最小コスト有向木を出力する．

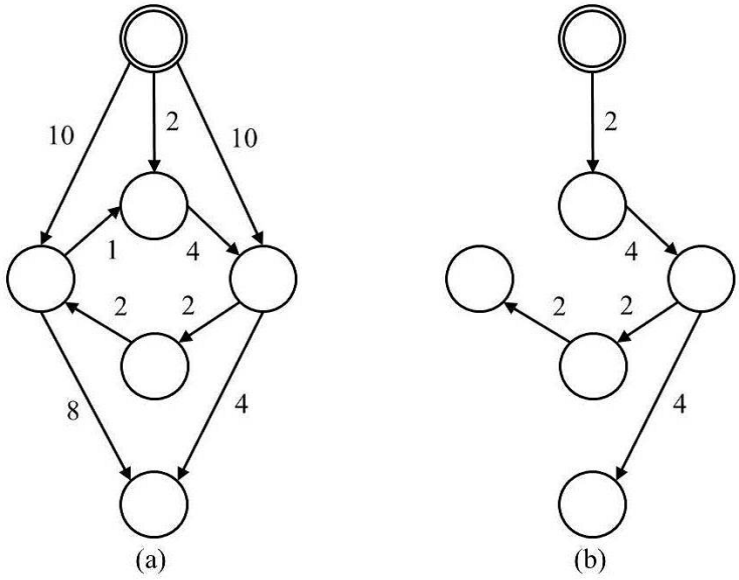

図 5　最小有向木問題の入力例．入力（a），入力例に対する最小コスト有向木 T (b)．二重丸の点は根 r，各枝にコストが付随している．

以下に有向木を特徴づける 1 つの命題を示す．この命題は多フェーズグリーディアルゴリズムにおいて重要な役割をもつ．

補題 6.1 $T = (V(T), E(T))$ は G の部分グラフであるとする．T が根 r に関する有向木であるとき，そのときのみ，T は有向閉路をもたず，かつ，任意の点 $v \in V(T)$ について v に向かう枝が 1 本のみ存在する．

補題 6.1 を証明する．T が根 r に関する有向木であるとする．すると T は有向木をもたない．さらに各点 v に一本のみ向かう枝が $E(T)$ に存在する．

一方，T に有向閉路をもたず，すべての点 $v \neq r$ に対して向かう辺が $E(V)$ に 1 本のみ存在するとする．T が有向木であることを示すには，根 r からすべての点 v に有向パスが存在することを示せばよい．v からはじめて，$E(T)$ の枝の向きと逆の方向に進むことを繰り返す．T は閉路をもたないので，それ以前に訪れたことにある点に戻ってくることはなく，動作は終了する．また，向かう枝をもたないのは点 r のみであるため，この動作は r に到着して終了する．この動作でたどった点が r から v へのパスになる．

6.3 多フェーズグリーディアルゴリズム

以下に最小コスト有向木問題の最適な有向木を求めるアルゴリズムを示す [6]．アルゴリズムの基本的な構造は，はじめに各点 v に向かう枝から 1 本のみを選び，そこから再帰的，逐次的に有向木を求める．

多フェーズグリーディアルゴリズム

• 有向グラフ $G = (V, E)$ の最適有向木 $T = (V, F)$ の探索．

For 各点 $v \neq r$ について

　　v に入る枝におけるコスト最小値を c_v とする．

End For

　　各点 $v \neq r$ においてコスト c_v の向かう枝を 1 本選び集合 F^* を作成．

If　F^* が有向木ならば，(V, F^*) を返す．

Else 有向閉路 $C \subseteq F^*$ が存在する．

　　C を 1 つのスーパーノードに縮約し, $G' = (V', E')$ を作成．

　　G' の最適有向木 (V', F') を再帰的に求める．

　　C の枝を 1 本除いてすべて加えることにより，(V', F') を (V, F) に拡張する．

Endif

このアルゴリズムは補題 6.1 を考慮し生成されている．はじめに各点 $v \neq r$ においてコスト最小値の向かう枝を 1 本選び，集合 F^* を作成する．集合 F^* で構成されるのが有向木なら，それが最小コスト有向木であり，そうでないときは有向閉路が存在することになる．有向閉路を C とすると，C に含まれる点を 1 つの点と考える．そのことをスーパーノードに縮約するという．スーパーノードに縮約し，新たに各点 $v \neq r$ においてコスト最小値の向かう枝を 1 本選ぶ．この動作を繰り返すことにより，最終的に有向木を 1 つ決定する．その有向木において，各 C から適切に 1 本除いてすべて加える．それによって G の最適有向木が求まる．

はじめに集合 F^* で構成されるグラフを図 6(a) とする．r の点以外は各点に向かう枝が 1 本選ばれる．図 6(a) は有向木ではなく，有向閉路が 2 つ存在する．それぞれを $C1$, $C2$ とし，スーパーノードに縮約する．スーパーノードに縮約すると，2 つの点に向かう枝は存在しないため，もう一度コスト最小値の枝が選ばれる．その枝を図 6(b) の s, t と仮定する．$C1$, $C2$ に含まれる点を 1 つの点と考えると，図 6(b) は有向木である．次に図 6(b) のグラフを拡張する．前述したように拡張とは C の枝を 1 本除き，そのほかの枝をすべて加えることである．除く枝に相当するのは図 6(b) の s'，t' である．したがって図 6(c) のような有向木を出力する．

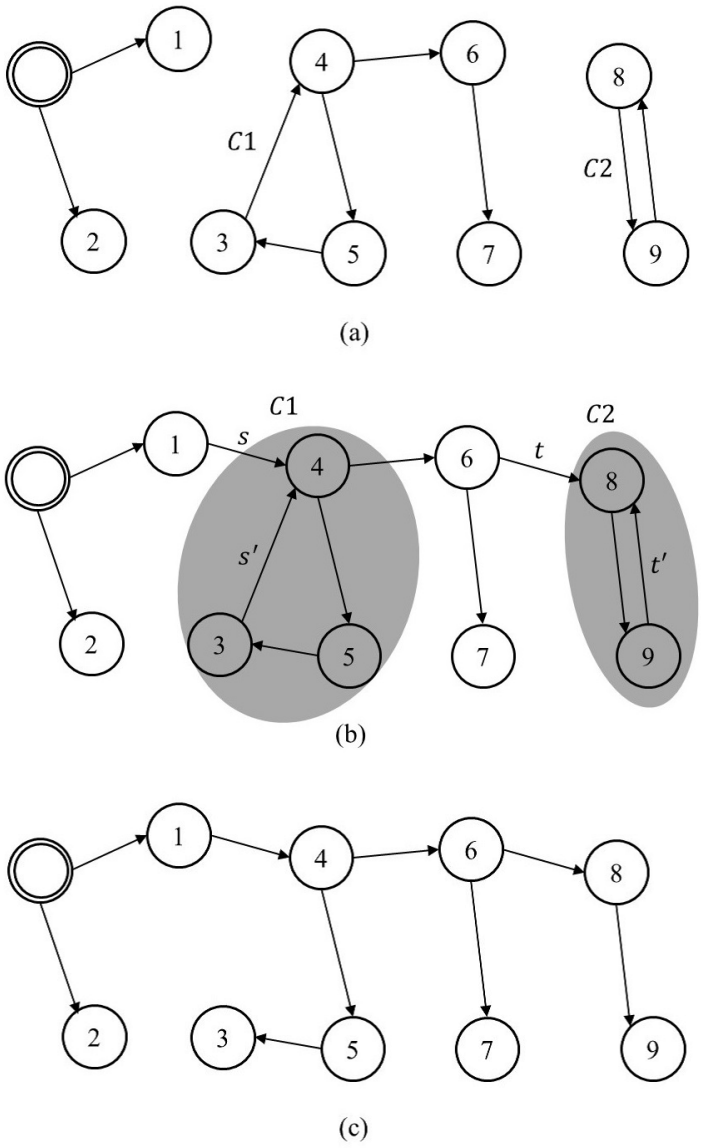

図 6　多フェーズグリーディアルゴリズムによる有向閉路の解消．ある入力から各点においてコスト最小値の向かう枝を 1 本選んだグラフ (a)，有向閉路 $C1$, $C2$ の解消 (b), 辺 s' を辺 s，辺 t' を辺 t に変換し，有向閉路を解消する．有向閉路を解消した有向木 (c)．

6.4　多フェーズグリーディーアルゴリズムの解析

入力の点数を n，枝数を m と仮定する．各点について最小コストの枝を選択するため，枝の総数は m であることから，その計算時間は，$\mathrm{O}(m)$ である．有向閉路が存在するとき，同様の動作を繰り返すことになる．有向閉路に含まれる点数は最小で 2 であり，最終的に根以外の点がすべて縮約されたと考える．最小コストの枝を選択する動作の回数は

高々$n-1$回である．よって全体の計算時間は

$$m(n-1) = \mathrm{O}(nm - m)\,, \tag{6.2}$$

である．したがって多フェーズグリーディーアルゴリズムは多項式時間での実装が可能である．

6.5 ヒューリスティクス (H2)

提案するヒューリスティクス (H2) を示す．ヒューリスティクス (H1) と同様に H2 も大きく分けて 2 段階からなる．第 1 段階は多フェーズグリーディーアルゴリズムに従い全域木を求める．第 2 段階では，得られた全域木から最適な部分木を求める．これは H1 と同様のアルゴリズムを用いる．以下に提案ヒューリスティクス (H2) を示す．

ヒューリスティクス (H2)

全域木の構築

1. 入力を有向グラフへ変換.
2. 多フェーズグリーディーアルゴリズムに従い，枝を選択.

点の削除

• H1 と同様に H2 の点の削除は最適な部分木を求めることと同意義である．最適な部分木を求めるアルゴリズムを後に示す.

第 1 段階は前述した有向グラフの変換のアイデアと多フェーズグリーディーアルゴリズムを用いる．H1 で用いたグリーディアルゴリズムは局所的かつ楽観的なアルゴリズムであった．多フェーズグリーディアルゴリズムはグリーディな形式をとっているが，枝を選ぶ規則が洗練されたものであり，最小コストである有向木が選ばれるという利点がある．それは，はじめに順序を決定して逐次的に決定しているためである．H1, H2 はともに第 2 段階に最適な部分木を求めるアルゴリズムを用いる．このアルゴリズムは自然なグリーディーアルゴリズムの 1 つであり，次章で本研究で提案する最適部分木に対して，最適な木を導くことを証明する．

7　最適部分木問題

有向木 $T = (V(T), E(T))$，有向枝 $e \in E(T)$ のコスト w_e，根 r が与えられる．r を含む部分木を T' とするとき，以下の目的関数

$$\sum_{e \in E(T')} w_e, \tag{7.1}$$

を最大にする木 T' を求めたい．

最適部分木問題の入力例を図 7(a) に示す．二重丸の点は根 r，各辺にコストが付随している．その入力に対して，実行可能な木を図 7(b) に示す．目的関数の値は 10 である．図 7(b) は入力に対して目的関数値を最小とする最適な木 T である．

7.1　最適な部分木を探索するアルゴリズム

上記の問題は入力である有向木から最適な部分木を求める問題である．その部分木 T を探索するアルゴリズムを以下に示す．

アルゴリズム 7.1

1. 根を始点とする深さ優先探索を行い，各点に後行順でラベル付け．
2. 後行順に，各点 v に対して，以下を実行．

- 点 v の親を u とし，u から v に向かう枝を e とする．
- 各点に $f_{i \in V(T)}$ を導入し，初期値を 0 とする．
- **If** $f_v - w_e > 0$,

  f_u を $f_u = f_u + f_v - w_e$ に更新．
- **Else** 点 v を含めたすべての子孫を削除．

アルゴリズム 7.1 は入力が有向木のとき，最適な部分木を求めることができる．また無向木に関しても，有向木に変換することでアルゴリズム 7.1 が適用できる．したがって入力が無向木である H1 に適用できる．

7.2　アルゴリズムの正当性

アルゴリズム 7.1 が最適な部分木を出力することを示すため，次の定理が成り立つことを証明する．

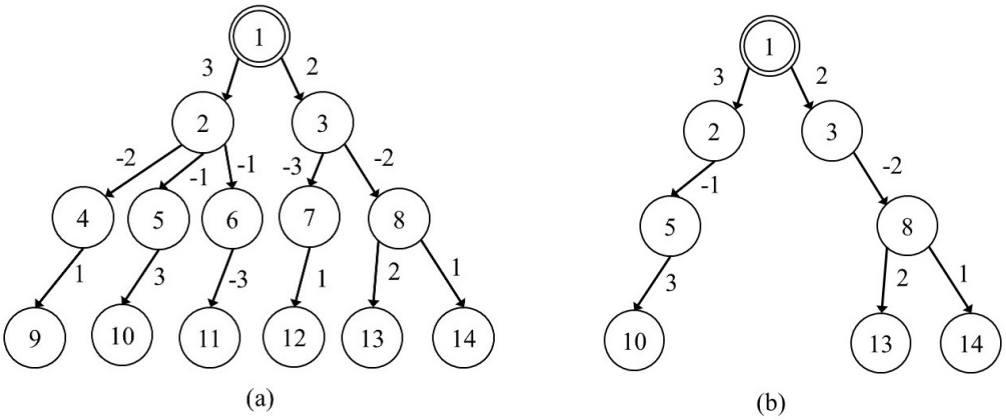

図 7　最適部分木問題の入力例．入力（a），入力例に対する最適な部分木 T (b)．二重丸の点は根 r，各辺にコストが付随している．

定理 7.2. アルゴリズム 7.1 は最適部分木問題に対して，最適な部分木を出力する．

定理 7.2 の正当性を帰納法で証明する．入力を深さ優先探索し，有向木の点にそれぞれの深さのラベルを付ける．深さとは根から最短で到達できるパスの数である．アルゴリズムで求めた部分木を T^* とし，深さ優先探索したときの最大の深さを n とする．ラベル n の点は葉と呼ばれ，子孫を持たない．ラベル n の点を根としたとき，ぞれぞれが最適な部分木である．

次にラベル $n-1$ の点を考える．その点は深さ n の点を子孫に持つか，持たないかどちらかである．子孫を持つとき，$n-1$ の点を根とした部分木を考える．部分木が最大の重みになるのは，根とその子孫を結ぶ枝が負でない枝を選択するときである．ただし枝のコストが 0 のとき，枝の選択の有無に関係なく最大の重みになる．アルゴリズム 7.1 は枝のコストが 0 のとき，その枝を削除する．はじめ各点のチャージは 0 であるため，$n-1$ の点とその子孫で構成される木は最適な部分木である．一方で子孫を持たないとき，ラベル n と同様である．

ラベル i $(2 \leq i \leq n-2)$ の点を考える．ラベル i の点を根としたとき，それらの子孫で構成される木は最適な部分木と仮定する．ラベル $i-1$ の点を根としたときも正しいことを示せば，ラベル 1 のときも最適な部分木であるといえる．ラベル i が根である部分木は最適であると仮定したため，ラベル i のチャージは最大である．ラベル $i-1$ とその子孫にあたるラベル i をつないでいる枝について，その枝のコストは点のチャージより大きい．ラベル i のチャージは最適であることから，ラベル $i-1$ を根としたときの部分木は最適といえる．

したがってラベル i の点について，それぞれ最適な部分木と仮定したとき，ラベル $i-1$ の点もそれぞれ最適な部分木である．そこからラベル 1 に相当する r のときも最適な部分木であり，木 T^* は最適な部分木といえる．以上から定理 7.2 は正しい．

7.3 計算時間の解析

入力の点数を n，枝数を m と仮定する．深さ優先探索による点のラベル付けは根を始点としているため，その計算時間は $\mathrm{O}(n+m)$ である．辺の削除について，辺の判定より削除した点は 2 度と削除されない．そのため各点における辺の削除は全体で $\mathrm{O}(n)$ である．よって全体で $\mathrm{O}(n+m)$ とできる．ここで全域木の性質から $m=n-1$ より，全体の時間計算量は $\mathrm{O}(n)$ である．したがってアルゴリズム 7.1 は多項式時間での実装が可能である．

8　計算実験

提案する 2 つのヒューリスティクスについてランダムグラフとベンチマークの入力を用いた性能検証を行う．実験環境は以下のとおりである．

Machine: lenovo ThinkPad X240
CPU: Intel(R) Core(TM) i3-4010U 1.70GHz
OS: Microsoft Windows 7 Enterprise
Memory: 4GB
Compiler: Microsoft Visual C++ 2008 Express Edition

ベンチワークとはあらゆる分野に存在し，アルゴリズムの性能を確認するための入力である．PCSP 問題にもベンチマーク問題は存在し，本研究では K, P, C, D, Cologne グループについて計算実験を行う．ランダムグラフにおける最適解は前述した整数計画問題の定式化から導く．整数計画問題を解くために非商用ソフト SCIP ソルバーを用いた．

8.1　点数を固定したランダムグラフに対する結果

ランダムグラフを点数 10 に固定し，辺数をいくつか設定して 2 つのヒューリスティクス H1，H2 の性能比較を行った．性能比較の対象に 3-近似アルゴリズムを用いた．各サンプル数を 100 ずつ生成し，それらのグラフ構造をランダムに設定する．ただし，各サンプルの点はすべて r へのパスが存在する．各点のペナルティ，各辺のコストはどちらも [1, 100] の一様乱数にした．

点数が 10 における近似比の結果を表 2 に示す．近似比とは，各ヒューリスティクスで求めた評価値を最適値で割った商である．縦軸はそれぞれ辺数，H1 の近似比，H2 の近似比，3-近似アルゴリズムで求めた近似比である．各近似比は，100 の試行を繰り返して算出した近似比の平均である．例えば，点数, 辺数がそれぞれ 10，20 のとき，H2 を用いた評価値による近似比の平均値は 1.65 である．H1 と H2 について，辺数の増加とともに，近似比は増加した．辺数の値にかかわらず，H2 の近似比は H1 より良好な結果を示した．その傾向は図 8 から，点数を固定したとき，辺数が多いほど H2 の近似精度は H1 より優れていた．一方，3-近似アルゴリズムは辺数の増加とともに，近似比が減少した．

表 2　H1, H2, 3-近似アルゴリズムから求めた近似比の平均．点数 10 に固定．各サンプル数は 100，縦軸に辺数と各近似比の平均．

$\|E\|$	15	20	25	30	35	40
H1	1.055	1.107	1.158	1.203	1.246	1.277
H2	1.045	1.065	1.081	1.123	1.115	1.156
3-近似	1.030	1.024	1.010	1.012	1.010	1.012

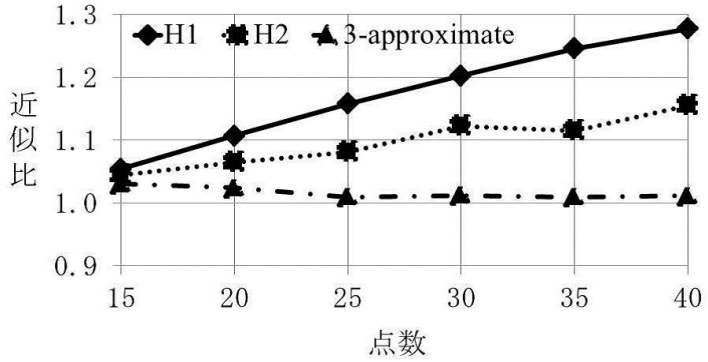

図 8　H1, H2, 3-近似アルゴリズムから求めた近似比の平均．点数 10 に固定．各サンプル数は 100．横軸に点数，縦軸に各近似比の平均．

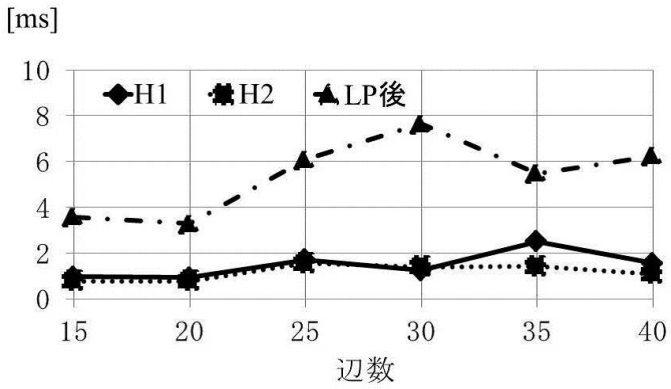

図 9　H1, H2, 3-近似アルゴリズムの LP 後による計算時間の平均 [msec]．点数 10 に固定．各サンプル数は 100．横軸に辺数，縦軸に各近似比の平均．

点数が 10 における計算時間を表 3 に示す．縦軸はそれぞれ辺数，H1, H2, 3-近似アルゴリズムの平均計算時間である．辺数に関わらず，H1 と H2 は同様に高速であることが確認された．一方 3 近似アルゴリズムは，点数，辺数がそれぞれ 10，20 のとき，平均の計算時間は 186.6mec である．この値は同条件における H1 と H2 に比べ，約 200 倍大きい．このほとんどは整数計画の緩和問題 (LP) を解く時間で占められる．LP 後に要した計算時間を図 9 に示す．3-近似アルゴリズムは LP を解く時間を考慮しなくても，H1 と H2 の計算時間に劣る．

表 3　H1, H2, 3-近似アルゴリズムの計算時間の平均 [msec]．点数 10 に固定，各サンプル数は 100．縦軸に辺数と各計算時間の平均．

$\|E\|$	15	20	25	30	35	40
H1	1.0	1.0	1.7	1.3	2.5	1.6
H2	0.8	0.8	1.6	1.4	1.4	1.1
3-近似	129.9	186.6	265.9	300.0	280.7	293.7

8.2　入力サイズの大きいランダムグラフに対する結果

大きな入力サイズに対して，2 つのヒューリスティクス H1，H2 の性能比較を行った．辺数を 1,000 に固定したときの評価値の結果を表 4 に示す．縦軸はそれぞれ点数，H1 の近似比，H2 の近似比である．H1 と H2 について，表 4 より点数の増加とともに評価値も増加した．近似比を比較すると，H2 が H1 より明らかに良好な結果を示した．

辺数を 1,000 に固定したときの計算時間を表 3 に示す．点数を 10 に固定したときと同様に，点数の増加に比例して H1 と H2 の計算時間は高速であることが確認された．H1 は図 10 より，点数の増加とともに線形関数的に計算時間は上昇している．そのため点数が多い現実問題に対して，高速に近似解を出力することが可能であると推測する．

表 4　大きな入力サイズに対する H1, H2 の評価値の平均，辺数 1,000 に固定．各サンプル数は 100．縦軸に点数，H1 の評価値の平均，H2 の評価値の平均．

$\|V\|$	100	200	300	400	500
H1	865	2,923	5,748	9,085	13,015
H2	633	2,421	5,120	8,563	12,661

表 5　大きな入力サイズに対する H1, H2 の平均計算時間 [msec]，辺数 1,000 に固定．各サンプル数は 100．縦軸に点数，H1 の計算時間，H2 の計算時間．

$\|V\|$	100	200	300	400	500
H1	865	2,923	5,748	9,085	13,015
H2	633	2,421	5,120	8,563	12,661

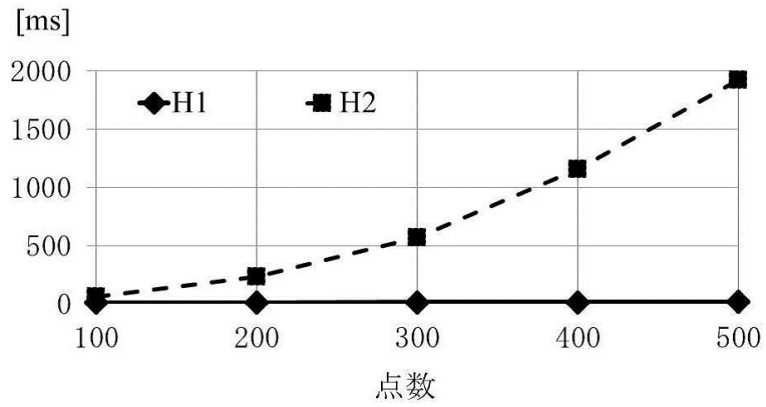

図 10　大きな入力サイズに対する H1, H2 の平均計算時間 [msec]，辺数 1,000 に固定．各サンプル数は 100．横軸に点数，縦軸に各計算時間．

8.3　K, P グループに対する結果

Johmson et al. [8] が計算実験で用いたベンチマークがある．それらのベンチマークには 2 つのグループがあり，その 1 つが K と呼ばれる．K グループは地図上の構造に良く似ている．入力の点を交差点と仮定しているため，ペナルティ 0 である点を多く持っている．もう 1 つは P と呼ばれ，構造はランダムである．P の特徴は総点数に占めるペナルティ 0 でない点の割合がおよそ 5 割であり，K と比べて 2 倍ほど多い．サンプル数は P が 11 つ，K が 23 つであり，最大の点数は 400，辺数は 1576 であった．これらのベンチマークは Lucena et al. [9] らや Canuto et al. [10] もヒューリスティクスの性能比較に用いている．

グループ K, P における近似比の結果を表 6 に示す．横軸はサンプル名，点数，辺数，H1 の近似比，H2 の近似比である．

近似比を比較すると，H1 が H2 と比べて良好な結果を示した．特に K に対する近似比について，H1 は H 2 よりおよそ 1,4 小さい結果を得た．これは P に比べて K が総点数に占めるペナルティ 0 である点の割合が多いことが関係していると推測する．

次にグループ K, P における各サンプルの結果について，表 7 に示す．横軸はサンプル

表 6　KP グループにおける H1, H2 の近似比の平均．横軸はサンプル，点数，辺数，H1 の近似比の平均，H2 の近似比の平均．

サンプル	$\|V\|$	$\|E\|$	H1	H2
K	248	920	1.129	1.133
P	245	745	1.266	1.351

名，点数，辺数，最適値 (OPT)，H1 の評価値，H2 の近似比，H1 の計算時間 (t_{H1})，H2 の評価値，H2 の近似比，H2 の計算時間 (t_{H2}) である．K に関して，点数 400 の入力が点数 100 の入力よりも近似比が 1.10 未満の結果を頻繁に得た．また H1, H2 の近似比の最大値が 1.60 を超えることはなかった．

表 7　K, P グループに対する H1, H2 の結果．OPT は最適値，t_{H1} は H1 の計算時間，t_{H2} は H2 の計算時間．太字は H1，H2 の近似比が 1.10 未満．サンプル数は 23.

	入力			H1			H2		
サンプル	$\|V\|$	$\|E\|$	OPT	評価値	近似比	t_{h1} [ms]	評価値	近似比	t_{ht} [ms]
K100.0	100	351	135,511	159,947	1.18	15	159,947	1.18	63
K100.1	100	348	124,108	149,330	1.20	16	149,330	1.20	46
K100.2	100	339	200,262	243,247	1.21	15	243,247	1.21	63
K100.3	100	407	115,953	144,349	1.24	15	144,349	1.24	78
K100.4	100	364	87,498	117,107	1.34	16	117,487	1.34	62
K100.5	100	358	11,9078	138,186	1.16	16	138,186	1.16	63
K100.6	100	307	132,886	157,041	1.18	16	159,892	1.20	62
K100.7	100	315	172,457	190,435	1.10	15	192,913	1.12	47
K100.8	100	343	210,869	227,220	**1.08**	15	227,369	1.08	62
K100.9	100	333	122,917	146,452	1.19	15	146,452	1.19	47
K100.10	100	319	133,567	152,423	1.14	15	152,423	1.14	62
K200	200	691	1,317,874	1,585,840	**1.09**	31	1,580,903	**1.09**	234
K400.0	400	1,515	350,093	398,802	1.14	31	398,802	1.14	4,524
K400.1	400	1,470	490,771	519,460	**1.06**	32	519,593	**1.06**	4,305
K400.2	400	1,527	477,892	516,754	**1.08**	31	518,443	**1.08**	4,212
K400.3	400	1,492	415,328	429,354	**1.03**	15	429,354	**1.03**	4,649
K400.4	400	1,426	389,451	433,079	1.11	32	433,079	1.11	4,431
K400.5	400	1,456	519,526	566,117	**1.09**	15	566,117	**1.09**	4,290
K400.6	400	1,576	374,849	397,427	**1.06**	15	398,755	**1.06**	4,477
K400.7	400	1,442	475,130	504,863	**1.06**	32	526,634	1.11	4,103
K400.8	400	1,516	418,614	441,332	**1.05**	15	440,804	**1.05**	4,649
K400.9	400	1,500	383,105	421,826	1.10	31	421,826	1.10	4,414
K400.10	400	1,507	395,848	423,987	**1.07**	32	423,987	**1.07**	4,353
P100.0	100	317	803,300	1,049,567	1.31	16	1,150,491	1.43	47
P100.1	100	284	926,238	1,167,810	1.26	15	1,420,186	1.53	62
P100.2	100	297	401,641	538,218	1.34	16	518,534	1.29	31
P100.3	100	316	659,644	876,136	1.33	31	938,286	1.42	31
P100.4	100	284	827,419	937,543	1.13	16	990,667	1.20	31
P200	200	587	1,317,874	1,585,840	1.20	31	1,580,903	1.20	96
P400.0	400	1,200	2,459,904	3,162,058	1.29	31	3,585,017	1.46	2,964
P400.1	400	1,212	2,808,440	3,626,618	1.29	47	3,555,709	1.27	1,981
P400.2	400	1,196	2,518,577	3,390,391	1.35	47	3,232,376	1.28	1,950
P400.3	400	1,175	2,951,725	3,443,999	1.17	46	3,894,680	1.32	2,512
P400.4	400	1,144	2,852,956	3,638,059	1.28	47	4,163,276	1.46	2,418

8.4 C, D グループに対する結果

Canuto et al. [10] は OR-Library[3] に存在する PCST の入力を用いて，新たに 80 つの入力を生成した．それらはグループ C, D に分類されていて，サンプル数は各 40 である．C の点数は 500，D の点数は 1,000 であり，ほとんどの点がペナルティ 0 である．C の辺数は 625 から 12500，D の辺数は 1,250 から 25,000 の間である．C, D について，コストは [1, 10] の一様乱数である。C, D は，さらに A と B に分類され，0 でない点のペナルティは，A が [1, 10]，B が [1, 100] の一様乱数に設定されている．

グループ C，D における近似比の結果を表 8，表 9 に示す．横軸はサンプル名，点数，辺数，H1 の近似比の平均，H2 の近似比の平均である．点数，辺数，近似比はともに，サンプルの試行から算出した平均値である．近似比を比較すると，C, D ともに H1 は H2 ち比べて良好な結果を示した．また C-B, D-B に分類された入力に関して，H1，H2 の近似比はどちらも 1.80 を超えた．

次にグループ C, D における各サンプルの結果について，表 10, 表 11 に示す．横軸はサンプル名，点数，辺数，最適値 (OPT)，H1 の評価値，H2 の近似比，H1 の計算時間 (t_{H1})，H2 の評価値，H2 の近似比，H2 の計算時間 (t_{H2}) である．C, D ともに H1, H2 の近似比の最小値と最大値の差が 3.00 以上と大きい値を得た．辺数もしくは点数が増加するごとに近似比の精度が下がっている傾向もみられた．

表 8　C グループに対する H1, H2 の近似比の平均，横軸はサンプル，点数，辺数，H1 の近似比の平均，H2 の近似比の平均．

サンプル	$\|V\|$	$\|E\|$	H1	H2
C-A	500	4,156	1.366	1.505
C-B	500	4,156	1.829	1.850
C	500	4,156	1.598	1.677

表 9　D グループに対する H1, H2 の近似比の平均，横軸はサンプル，点数，辺数，H1 の近似比の平均，H2 の近似比の平均．

サンプル	$\|V\|$	$\|E\|$	H1	H2
D-A	1,000	8,313	1.368	1.452
D-B	1,000	8,313	1.893	1.846
D	1,000	8,313	1.630	1.699

[3]OR-library: http://people.brunel.ac.uk/ mastjjb/jeb/info.html).

表 10　C グループに対する H1, H2 の結果．OPT は最適値，t_{H1} は H1 の計算時間，t_{H2} は H2 の計算時間．太字は H1，H2 の近似比の最小値と最大値．サンプル数は 40.

	入力			H1			H2		
サンプル	$\|V\|$	$\|E\|$	OPT	評価値	近似比	t_{H1} [ms]	評価値	近似比	t_{H2} [ms]
C1-A	500	625	18	27	1.50	16	27	1.50	1,731
C2-A	500	625	50	59	1.18	15	59	1.18	1,997
C3-A	500	625	414	439	1.06	16	439	1.06	1,732
C4-A	500	625	618	638	**1.03**	15	643	**1.04**	1,716
C5-A	500	625	1,080	1,116	**1.03**	15	1,232	1.14	2,013
C6-A	500	1,000	18	22	1.22	16	22	1.22	2,184
C7-A	500	1,000	50	59	1.18	16	59	1.18	2,839
C8-A	500	1,000	361	408	1.13	15	438	1.21	2,621
C9-A	500	1,000	533	572	1.07	31	625	1.17	1,872
C10-A	500	1,000	859	970	1.13	16	981	1.14	1,560
C11-A	500	2,500	18	27	1.50	31	27	1.50	3,993
C12-A	500	2,500	38	50	1.32	31	59	1.55	3,588
C13-A	500	2,500	236	309	1.31	16	321	1.36	3,120
C14-A	500	2,500	293	398	1.36	31	526	1.80	6,100
C15-A	500	2,500	501	688	1.37	31	651	1.30	5,350
C16-A	500	12,500	11	26	2.36	156	26	2.36	12,246
C17-A	500	12,500	18	39	2.17	156	56	3.11	12,324
C18-A	500	12,500	111	175	1.58	156	217	1.95	19,220
C19-A	500	12,500	146	224	1.53	156	256	1.75	6,442
C20-A	500	12,500	266	343	1.29	187	412	1.55	15,398
C1-B	1,000	625	85	130	1.53	16	122	1.44	1,684
C2-B	1,000	625	141	222	1.57	15	231	1.64	1,919
C3-B	1,000	625	737	866	1.18	16	895	1.21	1,607
C4-B	1,000	625	1063	1,165	1.10	16	1,251	1.18	1,810
C5-B	1,000	625	1528	1,638	1.07	15	1,672	1.09	2,012
C6-B	1,000	1,000	55	77	1.40	16	136	2.47	2,169
C7-B	1,000	1,000	102	196	1.92	15	228	2.24	2,839
C8-B	1,000	1,000	500	631	1.26	16	707	1.41	2,637
C9-B	1,000	1,000	694	854	1.23	15	844	1.22	1,778
C10-B	1,000	1,000	1,069	1,405	1.31	16	1,283	1.20	1,529
C11-B	1,000	2,500	32	78	2.44	31	100	3.13	3,978
C12-B	1,000	2,500	46	91	1.98	31	94	2.04	3,634
C13-B	1,000	2,500	258	491	1.90	16	349	1.35	3,074
C14-B	1,000	2,500	318	554	1.74	31	592	1.86	5,787
C15-B	1,000	2,500	551	981	1.78	31	729	1.32	4,477
C16-B	1,000	12,500	11	44	**4.00**	156	33	3.00	12,246
C17-B	1,000	12,500	18	57	3.17	156	76	**4.22**	12,293
C18-B	1,000	12,500	113	259	2.29	156	207	1.83	11,404
C19-B	1,000	12,500	146	308	2.11	140	232	1.59	7,129
C20-B	1,000	12,500	267	428	1.60	156	417	1.56	25,412

表 11　D グループに対する H1, H2 の結果．OPT は最適値，t_{H1} は H1 の計算時間，t_{H2} は H2 の計算時間．太字は H1，H2 の近似比の最小値と最大値．サンプル数は 40.

	入力			H1			H2		
サンプル	$\|V\|$	$\|E\|$	OPT	評価値	近似比	t_{H1} [ms]	評価値	近似比	t_{H2} [ms]
D1-A	500	625	18	27	1.50	63	27	1.50	20,249
D2-A	500	625	50	59	1.18	31	59	1.18	19,781
D3-A	500	625	807	844	1.05	31	846	1.05	13,775
D4-A	500	625	1,203	1,245	**1.03**	31	1,245	**1.03**	13,759
D5-A	500	625	2,157	2,225	**1.03**	31	2,325	1.08	11,279
D6-A	500	1,000	18	27	1.50	47	27	1.50	28,485
D7-A	500	1,000	50	59	1.18	47	59	1.18	25,350
D8-A	500	1,000	755	797	1.06	47	846	1.12	22,948
D9-A	500	1,000	1,070	1,158	1.08	31	1,248	1.17	19,703
D10-A	500	1,000	1,671	1,862	1.11	47	1,981	1.19	14,508
D11-A	500	2,500	18	27	1.50	124	27	1.50	32,760
D12-A	500	2,500	42	59	1.40	110	59	1.40	56,300
D13-A	500	2,500	445	580	1.30	109	762	1.71	48,781
D14-A	500	2,500	602	841	1.40	109	835	1.39	28,267
D15-A	500	2,500	1,042	1,402	1.35	109	1,271	1.22	18,252
D16-A	500	12,500	13	25	1.92	624	27	2.08	111,661
D17-A	500	12,500	23	50	2.17	577	59	2.57	102,666
D18-A	500	12,500	218	373	1.71	608	415	1.90	70,069
D19-A	500	12,500	306	482	1.58	593	558	1.82	120,387
D20-A	500	12,500	536	695	1.30	593	778	1.45	52,316
D1-B	1,000	625	106	170	1.60	15	174	1.64	23,090
D2-B	1,000	625	218	353	1.62	15	346	1.59	22,585
D3-B	1,000	625	1,509	1,706	1.13	32	1,805	1.20	15,820
D4-B	1,000	625	1,881	2,075	1.10	31	2,241	1.19	14,465
D5-B	1,000	625	3,135	3,329	1.06	31	3,405	1.09	13,347
D6-B	1,000	1,000	67	137	2.04	47	173	2.58	32,363
D7-B	1,000	1,000	103	255	2.48	47	292	2.83	28,970
D8-B	1,000	1,000	1,036	1,317	1.27	46	1,454	1.40	16,897
D9-B	1,000	1,000	1,420	1,744	1.23	47	2,023	1.42	24,216
D10-B	1,000	1,000	2,079	2,658	1.28	47	2,520	1.21	16,983
D11-B	1,000	2,500	29	68	2.34	109	78	2.69	38,400
D12-B	1,000	2,500	42	107	2.55	125	109	2.60	60,441
D13-B	1,000	2,500	486	850	1.75	109	1,010	2.08	52,841
D14-B	1,000	2,500	665	1,172	1.76	125	982	1.48	32,258
D15-B	1,000	2,500	1,108	2,033	1.83	109	1,360	1.23	21,568
D16-B	1,000	12,500	13	44	**3.38**	593	53	**4.08**	116,882
D17-B	1,000	12,500	23	74	3.22	593	86	3.74	96,247
D18-B	1,000	12,500	223	516	2.31	608	399	1.79	68,791
D19-B	1,000	12,500	319	707	2.22	593	555	1.74	127,572
D20-B	1,000	12,500	537	898	1.67	593	729	1.36	54,068

8.5　光ファイバーの設置計画に基づいた入力に対する結果

P. Bachhiesl et al. [11] が作成した現実問題のベンチマークがある．それらのベンチマークも 2 つに分かれ，Cologne1，Cologne2 と呼ばれる．サンプルは現実問題である光ファイバーの設置計画に基づいており，合わせて 29 つ存在する．それらは GIS のデータに基づいているため，点数と辺数が等しい入力が多い．しかし顧客の選択や価値を考慮しているため，各サンプルに違いがあり，Cologne1，Cologne2 はさらに 5 つに分類される．

グループ Cologne1，Cologne2 における近似比の結果を表 12，表 13 に示す．横軸はサンプル名，点数，辺数，H1 の近似比，H2 の近似比である．点数，辺数，近似比はともに，サンプルの試行から算出した平均値である．近似比を比較すると，Cologne1, Cologne2 ともに H1 の方が H2 より良好な結果を示した．

次にグループ Cologne における各サンプルの結果について，表 14 に示す．横軸はサンプル名，点数，辺数，最適値 (OPT)，H1 の評価値，H2 の近似比，H1 の計算時間 (t_{H1})，H2 の評価値，H2 の近似比，H2 の計算時間 (t_{H2}) である．各サンプルは枝数，点数ともに大きな違いがないため，サンプルにおける H1，H2 の近似比の結果に大きな差はなかった．Cologne1 ののサンプル i04 は H1 と H2 の近似比が等しくなる結果も得た．

表 12　Cologne1 グループに対する，H1, H2 の近似比．横軸はサンプル，点数，辺数，H1 の近似比，H2 の近似比．

サンプル	$\|V\|$	$\|E\|$	H1	H2
i01	768	6,332	1.079	1.148
i02	769	6,342	1.048	1.088
i03	771	6,342	1.050	1.061
i04	761	6,293	1.317	1.317
i05	716	6,296	1.166	1.193
Cologne1	766	6,321	1.119	1.150

表 13　Cologne2 グループに対する，H1, H2 の近似比．横軸はサンプル，点数，辺数，H1 の近似比，H2 の近似比．

サンプル	$\|V\|$	$\|E\|$	H1	H2
i01	1,819	16,743	1.165	1.239
i02	1,820	16,740	1.454	1.516
i03	1,825	16,762	1.289	1.364
i04	1,817	16,719	1.019	1.031
i05	1,826	16,794	1.140	1.182
Cologne2	1,821	16,752	1.213	1.266

表 14　Cologne グループに対する H1, H2 の結果．OPT は最適値，t_{H1} は H1 の計算時間，t_{H2} は H2 の計算時間．サンプル数は 29.

	入力			H1			H2		
サンプル	$\|V\|$	$\|E\|$	OPT	評価値	近似比	t_{H1} [ms]	評価値	近似比	t_{H2} [ms]
i101M1	748	6,332	109,272	109,272	1.00	110	109,272	1.00	28,407
i101M2	748	6,332	315,925	342,006	1.08	93	365,297	1.16	30,192
i101M3	748	6,332	355,625	410,628	1.15	78	457,829	1.29	30,482
i102M1	749	6,343	104,066	104,066	1.00	63	104,066	1.00	29,517
i102M2	749	6,343	352,539	365,595	1.04	78	375,535	1.07	28,676
i102M3	749	6,343	454,366	502,399	1.11	78	544,559	1.20	28,600
i103M1	751	6,343	139,749	139,749	1.00	62	139,749	1.00	28,852
i103M2	751	6,343	407,834	427,901	1.05	78	429,386	1.05	28,616
i103M3	751	6,343	456,126	501,786	1.10	63	515,638	1.13	28,671
i104M2	741	6,293	89,921	90,886	1.01	78	90,886	1.01	28,277
i104M3	741	6,293	97,149	157,610	1.62	62	157,610	1.62	29,881
i105M1	741	6,296	26,717	26,717	1.00	63	26,717	1.00	29,054
i105M2	741	6,296	100,270	111,555	1.11	63	112,315	1.12	28,686
i105M3	741	6,296	110,351	153,014	1.39	62	160,999	1.46	27,954
i201M2	1,803	16,743	355,468	359,774	1.01	421	359,774	1.01	433,080
i201M3	1,803	16,743	628,834	775,131	1.23	499	796,595	1.27	415,979
i201M4	1,803	16,743	773,398	967,042	1.25	453	1,111,955	1.44	412,558
i202M2	1,804	16,740	288,947	450,062	1.56	515	452,007	1.56	432,463
i202M3	1,804	16,740	419,184	595,261	1.42	421	633,242	1.51	441,231
i202M4	1,804	16,740	430,034	595,663	1.39	437	633,831	1.47	416,023
i203M2	1,809	16,762	459,919	608,614	1.32	421	619,906	1.35	412,886
i203M3	1,809	16,762	643,062	815,449	1.27	452	870,701	1.35	413,371
i203M4	1,809	16,762	677,733	863,983	1.27	531	942,031	1.39	414,039
i204M2	1,801	16,719	161,701	161,701	1.00	452	161,701	1.00	427,470
i204M3	1,801	16,719	245,287	252,096	1.03	484	256,681	1.05	441,561
i204M4	1,801	16,719	245,287	252,096	1.03	436	256,681	1.05	440,834
i205M2	1,810	16,794	571,031	578,726	1.01	406	580,883	1.02	446,533
i205M3	1,810	16,794	672,403	831,747	1.24	484	812,112	1.21	418,576
i205M4	1,810	16,794	713,974	835,184	1.17	452	942,309	1.32	415,554

9　考察

ランダムグラフと 5 つの K, P, C, D, Cologne のベンチマークを用いた計算実験の結果から，H1, H2 の近似精度や計算時間について考察する．

はじめにランダムグラフについてである．点数を 10 に固定したとき，点数を 100 以上にしたときののランダムグラフついてともに H2 が H1 の近似精度を上回った．10 点に固定したランダムグラフについて，辺数を 15, 20, 25, 30, 35, 40 に設定した．辺数が 40 に近づく，つまり増加するにしたがって，H1 と H2 の近似精度に大きな差が出た．これは H1 で用いたグリーディアルゴリズムに弱点があると考察する．グリーディアルゴリズムは局所的に最適な辺を選ぶが，全体の最適化を目指すことと一致しない場合がある．一方，H2 は有向グラフに変換するステップを踏むが，その有向グラフから求められる有向木は最小コストである．

比較対象とした既存の 3 近似アルゴリズムは点数 10 のランダムグラフにおいて，H1, H2 よりも優れた近似精度であった．しかし計算時間は H1,H2 と比べて約 200 倍もの時間を要した．この計算時間のほとんどを整数計画の緩和問題で占め，サイズの大きいグラフに対して，緩和問題を解くことができないのが欠点である．本研究の実験環境では点数 15 以上の入力について，近似解を求めることができなかった．

サイズの大きいランダムグラフにおいて，H1，H2 は高速に近似解を出力した．H1 は線形関数的に上昇しており，現実問題のような入力に対しても高速に実装できると考察する．評価値は点数を 10 に固定したときと同様で，H2 の近似精度が良い結果を得た．

次に 5 つのベンチマークについて考察する．各グループの近似比の平均はすべて H1 が良好であった．これは入力がランダムグラフのときと異なる結果である．この結果が得られたのは入力グラフの構成方法が関係していると推測する．ランダムグラフでは，各点に [1, 100] の範囲の一様乱数によってペナルティを付随させた．したがってペナルティ 0 の点はほとんど存在しない．一方各グループで偏りはあるが，5 つのベンチマークは約 50%以上の点がペナルティ 0 に設定されている．特に現実問題に基づいている K は 75%以上，Cologne は 95%以上とペナルティ 0 の点の割合が多い．具体的に K ならば交通網の交差点，Cologne ならば光ファイバーの経由地点などをペナルティ 0 と仮定するためである．そしてペナルティ 0 の割合が多いと H2 で問題が生じる．有向グラフに変換するアイデアがほとんど機能せず，各辺が負のコストをもつことと同意義である．そのためランダムグラフのときと異なる結果を得たと考察する．

10　おわりに

本研究はPCST問題に対する2つのヒューリスティクスを提案した．また近似比と計算時間の観点から計算実験を行い，ヒューリスティクスの性能を比較した．計算実験から2つのヒューリスティクスの特徴について以下のことが挙げられる．

H1は交差点や経由地点を多くもつ入力において効果的である．これはH1のグリーディアルゴリズムが関係している．局所的に最適な選択を選ぶことを繰り返すことが，最適な全域木を構築することにつながっている．

H2は有効な点の選択肢が多い入力において効果的である．そういった入力は有向グラフに変換するアイデアが有能であり，そこから全体で最小コストの有向木が選択できる．

既存手法にはプライマルデュアル法に基づいたアルゴリズムもある．今後の課題として，この手法と比較し，H1, H2の新たな特徴を考察することが挙げられる．

参考文献

[1] D.P. Williamson and D.B. Shmoys, The Design of Approximation Algorithms, Cambridge University Press, 2011.

[2] 浅野孝夫, 近似アルゴリズム, 丸善出版, pp.212–214, 2014.

[3] A. Archer, M. Bateni, M. Hajiaghayi, and H. Karloff, "Improve approximation alogorithms for prize-collecting Steiner tree and TSP", SIAM Journal on Computing, vol.40, no.2, pp.309–332, 2011.

[4] S.A. Canute, M.G.C. Resende, and C.C. Ribeiro, "Local search with perturbations for the prize-collecting Steiner tree problem in graphs", Network, vol.38, pp.50–58, 2001.

[5] G.W. Klau, I. Ljubic, A. Moser, P. Mutzel, P Neuner, U. Pferschy, and R. Weiskicher, "Combining a memetic algorithm with integer programming to solve the prize-collecting Steiner tree problem", Proceedings of the Genetic and Evolutionary Computation Conference, vol.3102, pp.1304–1325, 1998.

[6] 浅野孝夫, 浅野泰仁, 小野孝男, 平田富男, アルゴリズムデザイン, 共立出版, 2008.

[7] M.X. Goemans, and D.P. Williamson, "A general approximation technique for constrained forest problems", SIAM Journal on Computing on Computing, vol.24, no.2, pp.296–317, 1995.

[8] D.S. Johnson, M. Minkoff, and S. Phillips, "The prize-collecting Steiner problem: Theory and practice", In Proceedings of 11th ACM-SIAM Symposium on Discrete Algorithms, pp.760–769, 2000.

[9] A. Lucena, and M.G.C. Resende, "Strong lower bounds for the prize-collecting Steiner problem in graphs", Discrete Applied Mathematics, vol.141, pp.277–294, 2004.

[10] S.A. Canuto, M.G.C. Resende, and C.C. Ribeiro, "Local search with perturbations for the prize-collecting Steiner tree problem in graphs", Networks, vol.38, pp.50–58, 2001.

[11] P. Bachhiesl, M. Prossegger, G. Pauius, J. Paulus, J. Werner, and H. Stoger, "Simulation and optimization of the implementation costs for the last mile of fiber optic networks", Network and Spatial Economics, vol.3, no.4, pp.467–487, 2003.

謝辞

本修士論文は，筆者が法政大学大学院理工学研究科システム工学専攻修士課程在学中の研究をまとめたものです．本研究に関して指導教員の五島洋行教授から，丁寧かつ熱心なご指導を賜りました．ここに感謝の意を表します．また，本研究を終始ご指導頂いた千葉英史専任講師，論文をご精読頂いた安田教授に深謝致します．そして日常の議論を通じて多くの知識や示唆を頂いた同期や同ゼミの後輩の皆様に感謝致します．